

Саша Окунев

Руководство по Figma

Книга выпущена в рамках обучающего проекта



slashdesigner.ru

[@slashdesigner](https://www.instagram.com/slashdesigner)

© Саша Окунев, 2019. Авторские права защищены.

Версия 1.1. Beta.

Эта книга бесплатна

Её актуальную версию всегда можно скачать на сайте slashdesigner.ru.

Закачай её в айпад

Книга адаптирована для чтения на планшетах.

Поддержи проект

Ты можешь сделать добровольное пожертвование на развитие проекта **/designer** любым удобным способом: [PayPal](#) или [Я Соберу](#).

Сообщество и обратная связь

Вопросы про Фигму можно задавать в [Фигма-чате](#).

На вопросы про интерфейсы отвечаю в [Яндекс Знатоках](#).

Оставь публичный отзыв о книге на странице [Отзывы и предложения](#).

Если найдёшь опечатку — присылай скриншот: [@okunev](#).

Changelog

Версии книги:

- **27 февраля 2019 - 1 Beta**

Первая публичная версия. Вошли 23 главы, 256 стр.

- **5 марта 2019 - 1.1 Beta**

Минорный апдейт. Исправлены опечатки первой волны, о которых написали читатели. Добавлены некоторые клавиши для Windows и новые скриншоты. Исправлен стиль нумерованных списков.

Что это и кому	10
О клавишах	11
1. Установка Фигмы	12
Клиенты	12
WebGL	12
Локальные шрифты	13
Практика: сетап	13
2. Файлы, проекты и команды	14
Импорт	14
Команды	15
Создание проекта	17
Проект для изучения: как скачать	18
Практика: первый файл	19
3. Сверху: тулбар и тёмная панель	20
Главное меню	21
Тулбар	22
Скрытие интерфейса	23
Переименуем файл	24
Как давать доступ по ссылке	25
Практика: тулбар и ссылки на проект	26
4. По центру: рабочая область	27
Фон рабочей области	29
Режимы отображения рабочей области	31
Линейка	34
Практика: рабочая область	35
5. Слева: панель слоёв и страницы	36
Страницы	38
Перемещение слоёв между страницами	39

Практика: список слоёв и страницы	40
6. Справа: панель свойств	41
Общие свойства слоёв	44
Практика по свойствам слоя	45
7. Фреймы	46
Фоновый цвет фрейма	47
Фреймы: Работа с сеткой	48
Шаг сдвига по сетке	50
Фреймы для профи	51
8. Шейпы	52
Прямоугольник	53
Чем отличаются фреймы и прямоугольники	54
Сдвиг и масштабирование	55
Заливка и обводка шейпов: первый взгляд	56
Прямоугольник и сетка	59
Плющим несколько фигур	59
Сдвигаем и липнем к объектам	60
Opt для определения расстояний	61
Ещё один способ мерить расстояния	62
Режим редактирования	63
Окружность, овал и пайчарт	71
Треугольник	76
Звезда	77
Линия	79
Стрелка	81
9. Перо	85
Частая ошибка: изломы линий	91
Минимализм и точность	92
Shift и Snap to Geometry: чертим прямоугольник пером	95

10. Векторные сети	100
Как сделать векторную сеть	103
Заливаем фрагменты сети	105
Bend Tool, Cmd	106
Рисуем иконки	108
11. Булевы группы и флэтен	110
Subtract на практике: вычитаем круги	113
Union на практике: соединяем шейки	116
Intersect на практике	117
Exclude на практике	118
Сходства и различия булевых групп в Фигме и Скетче	122
Принцип инверсии в Скетче	126
Вложенные булевы группы в Фигме	127
Flatten составных фигур	128
12. Режим Outline: векторные контуры	131
Находим невидимые шейки	132
Как выделять труднодоступные слои	132
13. Режимы цветового кодирования	134
HEX: Шестнадцатиричные цвета	134
RGB-цвета	135
RGBA-коды	136
HSB-цвета	137
14. Заливка и градиенты	139
Режим заливки Solid: ровный цвет	142
Режим заливки Linear: линейный градиент	144
Режим заливки Radial: радиальный градиент	150
Режим заливки Angular: угловой градиент	157
Режим заливки Diamond: делаем блики	163

15. Режим заливки Image	167
Режим Image / Fill	171
Режим Image / Fit	173
Режим Image / Crop	174
Режим Image / Tile	179
16. Цветокоррекция	181
Exposure	183
Contrast	184
Saturation	185
Temperature	187
Tint	188
Highlights	189
Shadows	190
17. Обводка	191
Отличия пунктирной обводки в Скетче и Фигме	194
Стили изгибов обводки	196
Типы окончания линии	197
18. Маски	198
Кадрирование фото	198
Альфа-маска	203
19. Адаптивность и ограничители	205
Ограничители шейпа по умолчанию	207
Как сделать резиновую шапку	211
Как сделать сайдбар с резиновой высотой	215
Как растянуть фоновое фото на весь фрейм	218
Как зафиксировать модальное окно по центру экрана	220
Как сделать так, чтобы при растягивании макета всё не ехало	223
20. Текстовые слои	224

Шрифты из Google Web Fonts	225
Auto Resize: Width	229
Auto Resize: Height	230
21. Выравнивание и распределение	233
Align: Выравнивание	233
Distribute: Распределение	234
22. Стили цветов	236
Копируем стили с одного объекта на другой	239
Сходства и различия стилей в Фигме и Скетче	240
Цвета можно называть и комментировать	241
23. Компоненты	242
Сделаем компонент кнопки	242
Немного о клавишах	244
Сравнение символов Скетча и компонентов Фигмы	245
Различие в логике дублирования	247
Различие в логике детача	248
Различие в реализации оверрайдов	249
Переход к мастеру	252
Лучшие практики работы с компонентами	253
Практика: компоненты	254
Об авторе	255

Что это и кому

Я посвящаю этот учебник любопытным самоучкам. Моя цель в том, чтобы вдохновить и прокачать дизайнеров, дав знания об удобном и мощном инструменте.

В книге в компактном виде собран солидный набор технических знаний, который позволит эффективно создавать интерфейсы.

Я подробно анализирую основные функции, а также их аналоги в Скетче. Разбираю реализацию символов и компонентов, стилей и ограничителей. Прежде всего ориентируюсь на тех, кто уже имел опыт работы в Скетче или Фотошопе.

Начинающие найдут здесь для себя правильный технический фундамент дальнейшего развития в дизайне: как использовать перо, что такое градиенты, режимы наложения и цветового кодирования. Продолжающие смогут быстро перестроиться на новый редактор.

Монополия Скетча и Маков закончилась. Расцвет Фигмы — революция в индустрии дизайн-инструментов: теперь если у человека есть компьютер, в котором заводится браузер, ему доступен потрясающе мощный инструмент для создания интерфейсов.

Бери эту книгу, изучай и твори.

О клавишах

Особо трепетное отношение я проявляю к горячим клавишам, потому что это мой метод изучения любого софта.

Я помечаю клавиши красным, чтобы они были заметны.

Поскольку Мак — всё ещё стандартный выбор дизайнера, все горячие клавиши я привожу для него, например **Command + Y**. **Option + 2**.

Названия клавиш сокращаю:

Command → **Cmd**

Option → **Opt**

Control → **Ctrl**

В Windows вместо **Cmd** используется **Ctrl**, а вместо **Opt** используется **Alt**. Некоторые сочетания клавиш для Windows вообще не похожи на маковские, о них буду рассказывать отдельно в следующих версиях этой книги.

1. Установка Фигмы

Клиенты

У Фигмы есть клиент для Mac и Windows, а также бесплатное приложение **Figma Mirror** для iOS и Android. Миррор — обязательный инструмент дизайнера мобильных приложений и сайтов.

Всё это можно скачать на официальном сайте: figma.com/downloads

WebGL

Также Фигму можно использовать в браузере, если тот поддерживает технологию WebGL. Разработчики Фигмы рекомендуют использовать браузер [Google Chrome](https://www.google.com/chrome/).

Проверить, поддерживает ли браузер WebGL можно на webglreport.com.

Если возникли проблемы с отображением в браузере, обратиться к руководству пользователя: [Как включить WebGL](#) (англ.).

Локальные шрифты

Фигма может использовать шрифты, установленные в операционной системе как в редакторе, так и в браузере.

Если используется клиент для Мака, после установки нового шрифта он не виден в шрифтах. Чтобы это исправить, нужно выйти и заново войти в приложение Фигмы. Чтобы шрифты отображались в браузере, нужно приложение **Font Helper**. Ссылка на его скачивание доступна в меню **Help and Account** → **Account Settings**.

Либо по ссылке: figma.com/settings

Font Helper должен постоянно быть в памяти во время работы.

Практика: сетап

1. Установить клиент Фигмы.
2. Зарегистрировать бесплатный аккаунт.
3. Войти в Фигму в браузере.
4. Задание на пятёрку: Найти в настройках, как привязать номер телефона к аккаунту Фигмы, и включить двухфакторную авторизацию. Это сильно затруднит злоумышленникам возможность взломать твой аккаунт в Фигме.



Files



Projects



Teams

2. Файлы, проекты и команды

Файлы — аналог скетч-проектов. В них можно создавать макеты.

По умолчанию создаются в общей категории **Drafts**. На файлы можно давать [такие ссылки](#). Также файлы можно выгружать из Фигмы в формате **.fig**. Можно создать неограниченное количество файлов.

Проекты группируют файлы. Также файлы можно создавать сразу в проектах. В бесплатном аккаунте может быть до трёх проектов.

Команды — группы проектов, в которые даём доступ участникам.

Пример: Есть команда **/designer**, в ней есть проект **Figma Guide**, а в нём есть файл **Figma-Guide-Rus**.

Также есть секция **Deleted Files**, где сохраняются удалённые файлы. Там же можно удалять файлы безвозвратно.

Импорт

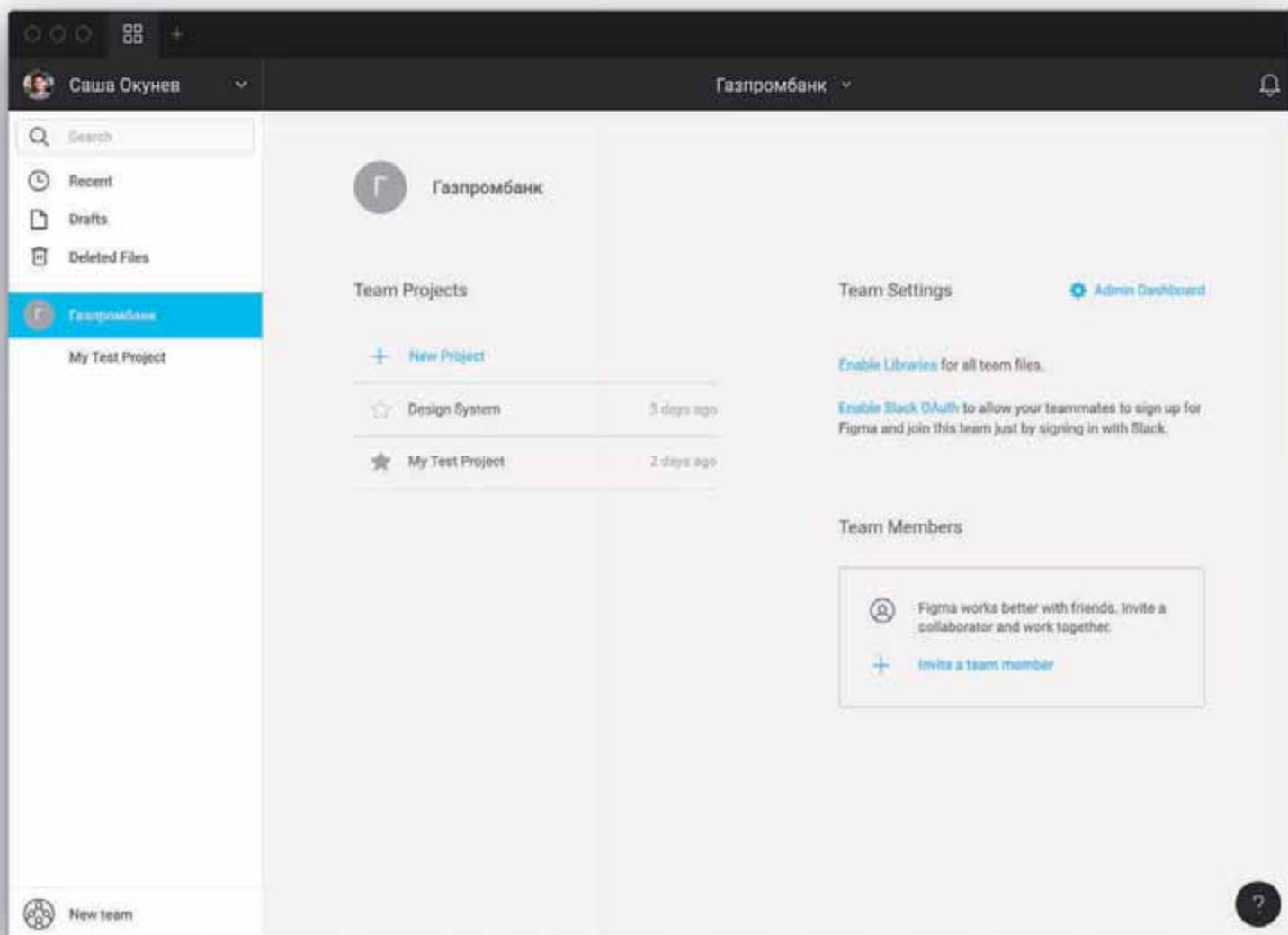
В Фигму можно импортировать файлы в разрешениях **.fig** или **.sketch**. Чтобы импортировать, нажимаем кнопку **Import**, либо перетаскиваем файлы в окно Фигмы.

Команды

Команды нужны, чтобы организовать работу группы людей, которые работают над несколькими проектами.

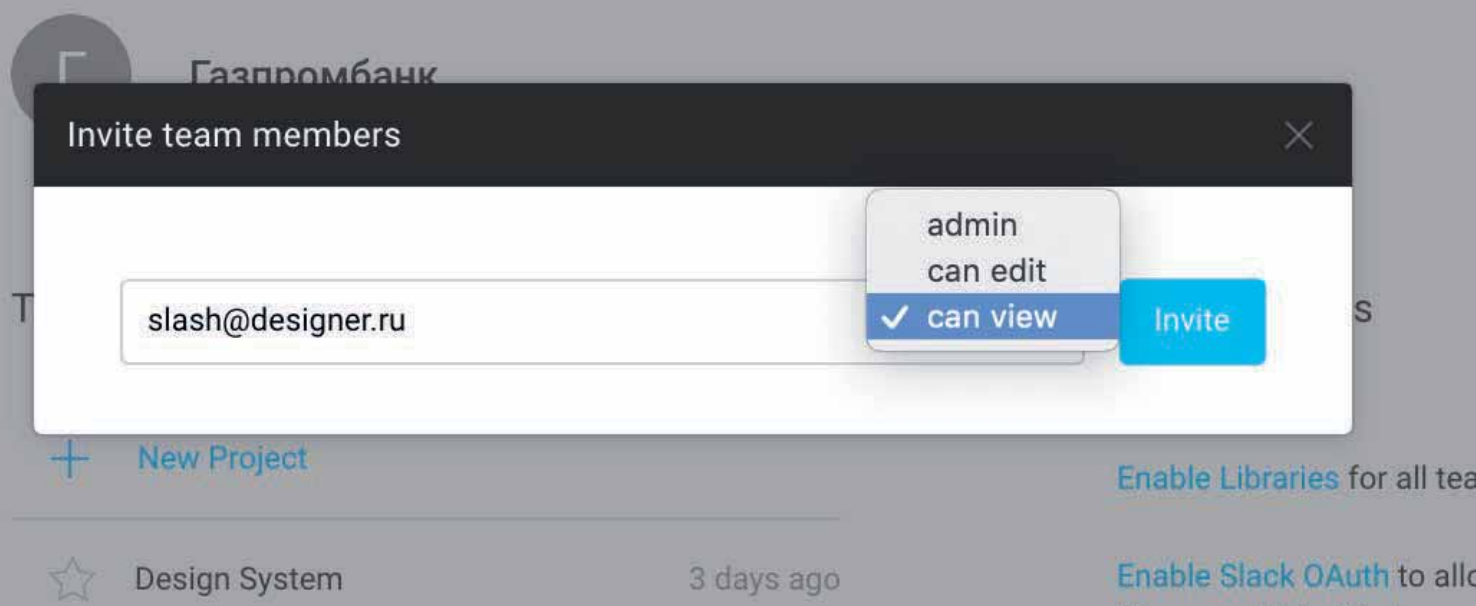
Чтобы создать команду, нажимаем **New team**.

На странице команды доступны проекты, которые к ней относятся, а также настройки.



Приглашаем команду

Нажимаем **Invite a team member** и определяем уровень доступа. За каждого редактора или администратора команды Фигма требует 15\$ в месяц. Количество зрителей неограниченно.



У приглашённого пользователя в его окне Фигмы в верхнем правом углу появляется уведомление, которое можно принять или отклонить.

Enable Libraries

Функция **Enable Libraries** позволяет сделать доступными библиотеки для всех файлов команды.

Slack OAuth

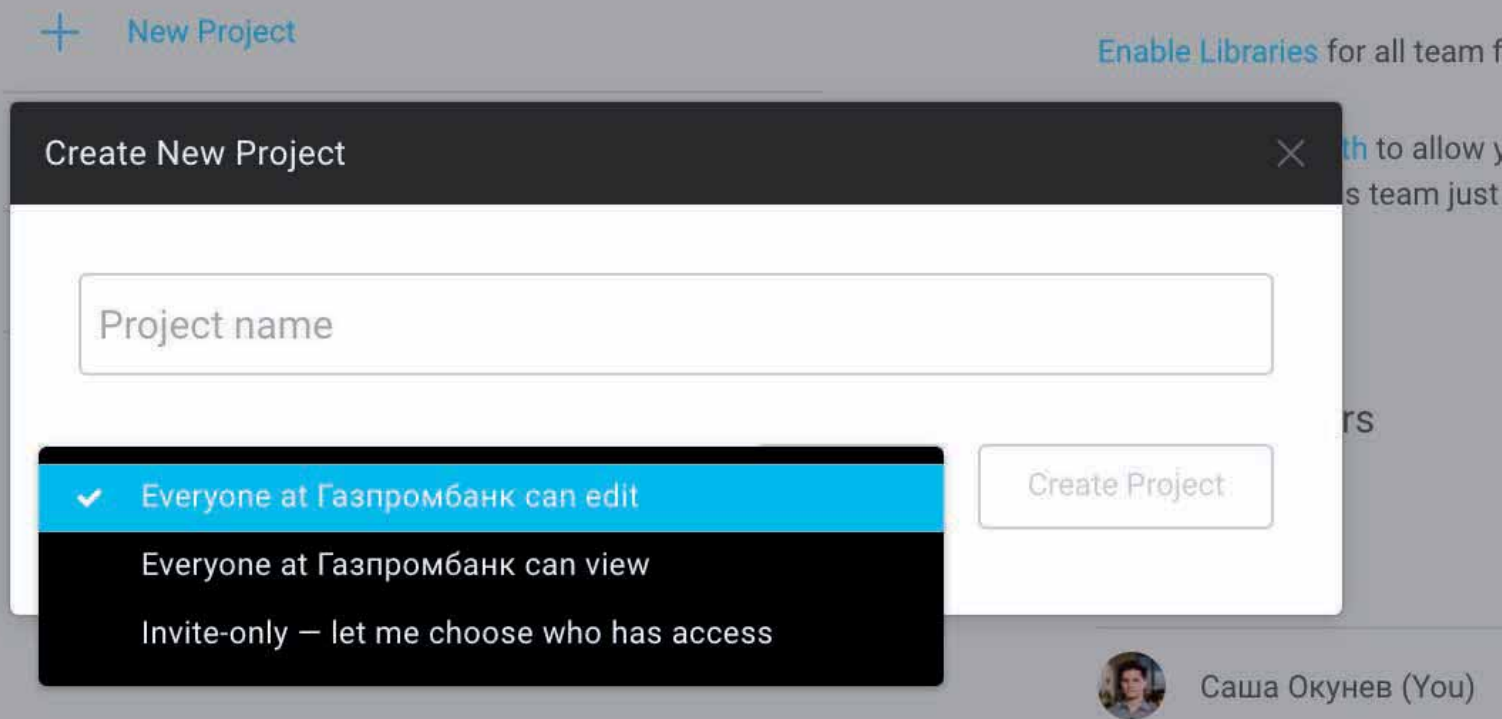
Функция **Enable Slack OAuth** позволяет авторизоваться в Фигме и присоединиться к команде, если авторизован в Слаке.

Создание проекта

На странице команды мы можем создавать проекты, нажав **New Project**, **Cmd + N**.

При создании проекта можно определить **уровень доступа**:

- **Everyone at Team can edit** – любой член команды может редактировать файлы внутри неё.
- **Everyone at Team can view** – изначально все могут только смотреть файлы.
- **Invite-only** – доступ в проекты по индивидуальным приглашениям.



Проект для изучения: как скачать

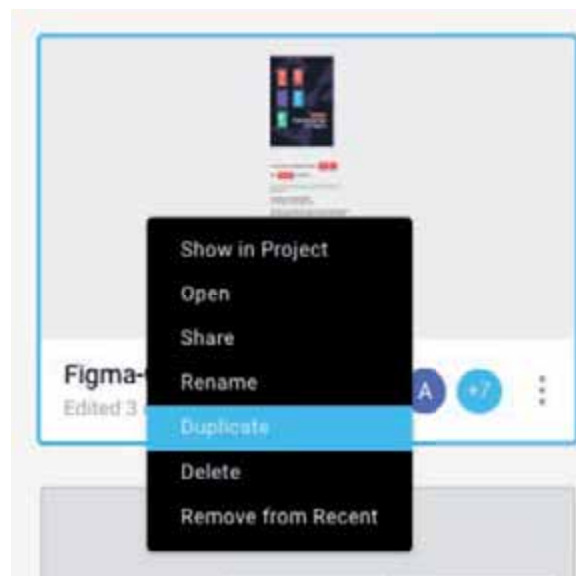
В процессе написания этой книги я делал иллюстрации в Фигме и снимал скриншоты, работая в проекте [Figma-Guide-Rus](#).

К нему можно получить доступ по ссылке bit.ly/figma-examples.

Вариант 1, через дублирование файла

Когда ты прошёл по ссылке, проект появляется во вкладке **Recent**.

Там его можно дублировать в своё пространство. Копия будет доступна для редактирования.

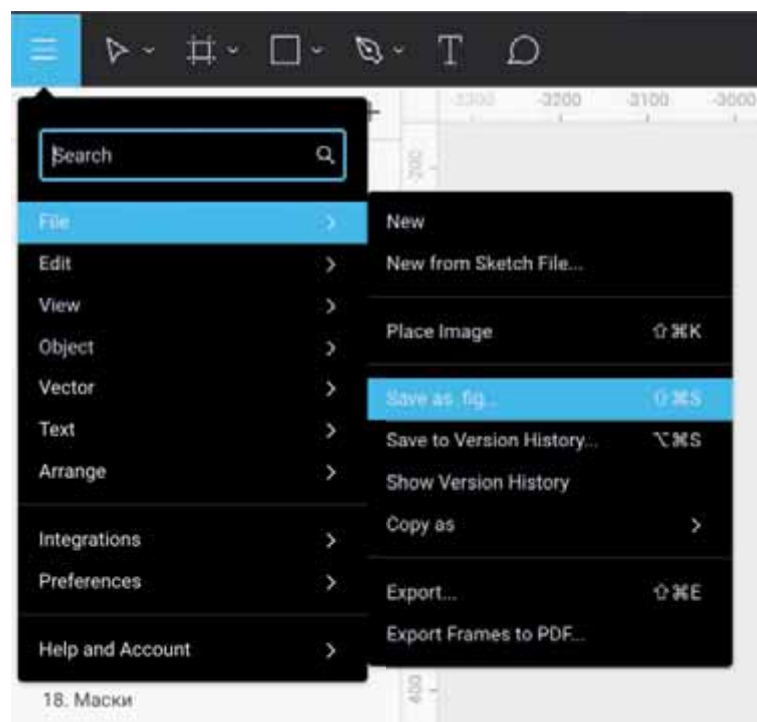


Вариант 2, с сохранением .fig-файла

Его можно открыть в Фигме, сохранить как .fig-файл и скопировать в свой аккаунт для дальнейшего разбора:

File → **Save as fig...**, **Shift + Cmd + S**.

В каждой главе учебника я даю ссылку на соответствующие страницы этого проекта.



Практика: первый файл

1. Открыть десктопный клиент Фигмы.
2. Создать команду, выбрать для неё аватар.
3. Создать в ней проект.
4. Создать в нём первый файл.

3. Сверху: тулбар и тёмная панель

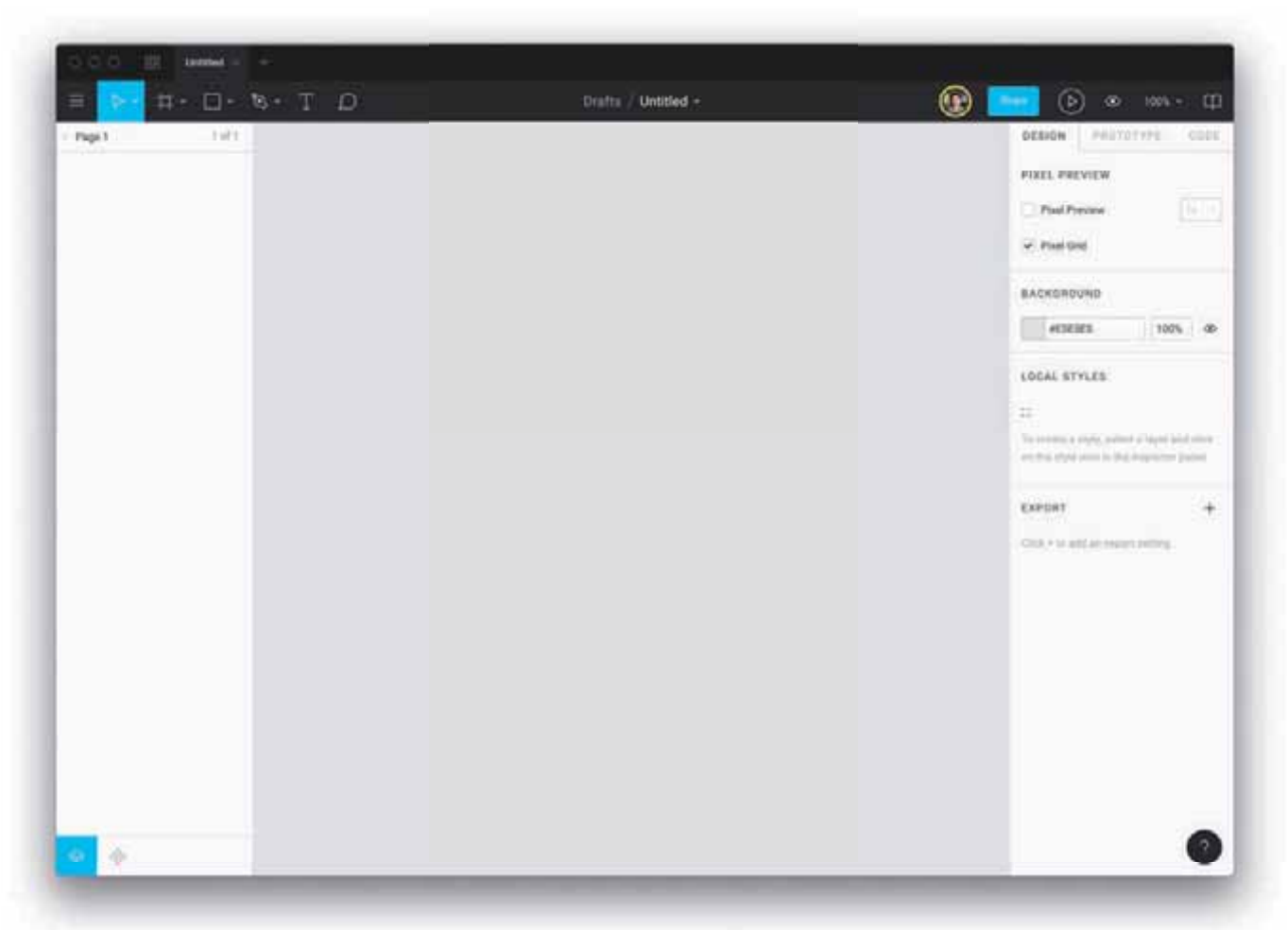
В этой главе рассмотрим главное меню и тёмную верхнюю панель, в которой мы выбираем инструменты и настраиваем доступ.

Открыли новый файл. Закреть — **Cmd + W**. Слева от открытой вкладки открытого файла видна вкладка возврата на страницу проекта. Можно создать ещё один файл, нажав **+**. Откроется ещё одна вкладка.

Переключаться между вкладками можно как в браузере:

Shift + Cmd + [— Предыдущая вкладка.

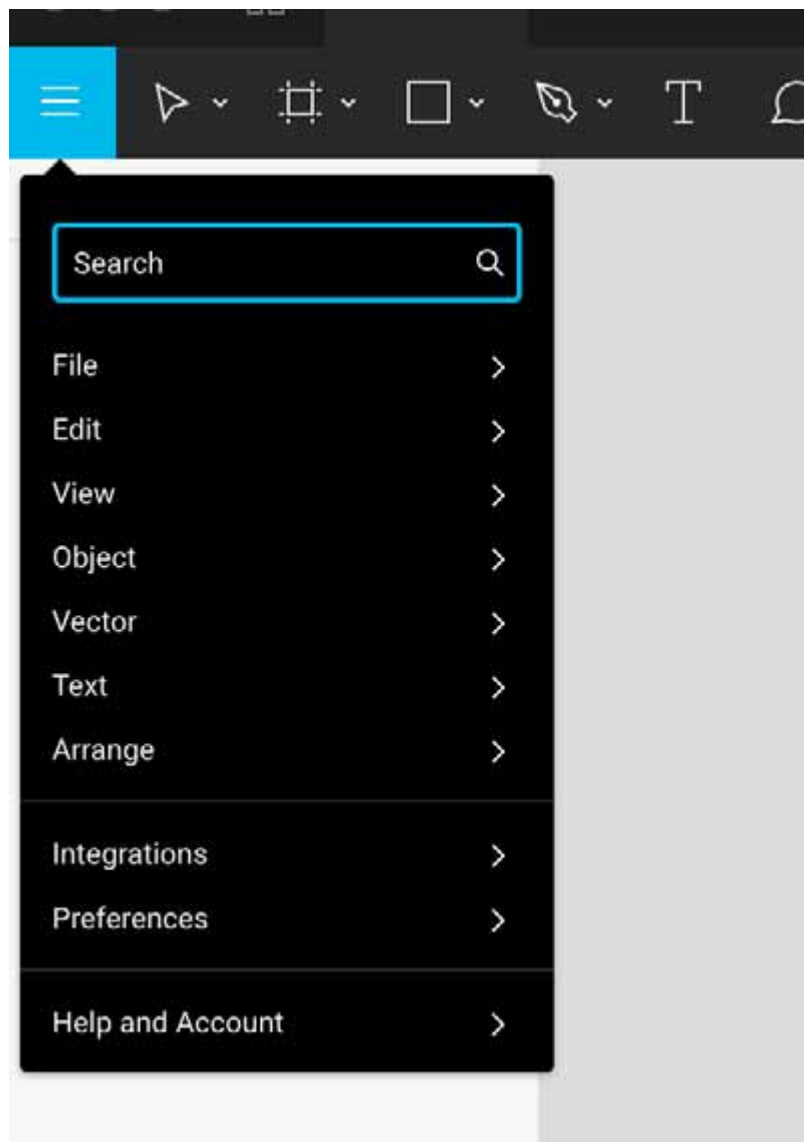
Shift + Cmd +] — Следующая.



Главное меню

В верхнем левом углу — главное меню программы, которое можно открыть клавишей **Cmd + /**.

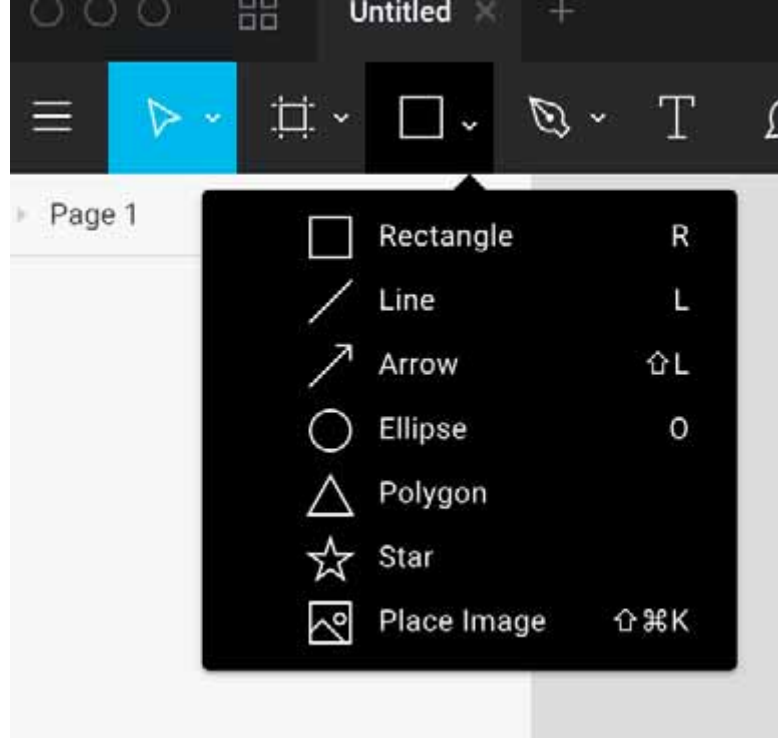
В меню есть поиск по пунктам. Развернув меню клавишей, можно продолжить печатать нужную команду.



Тулбар

Панель инструментов в Фигме (**Toolbar**) работает так же как в других редакторах. Мы разберём каждый инструмент в дальнейших главах.

Изначально активен инструмент **Move**. Если выбрать другой, в **Move** можно вернуться, нажав **Esc**.



V, Esc **M**ove

K Scale

F, A **F**rame

S **S**lice

R **R**ectangle

L **L**ine

Shift + L **L**ine with Arrow

O Ellipse, **O**val

P **P**en

Shift + P **P**encil

C **C**omment

Shift + Cmd + K Place Image

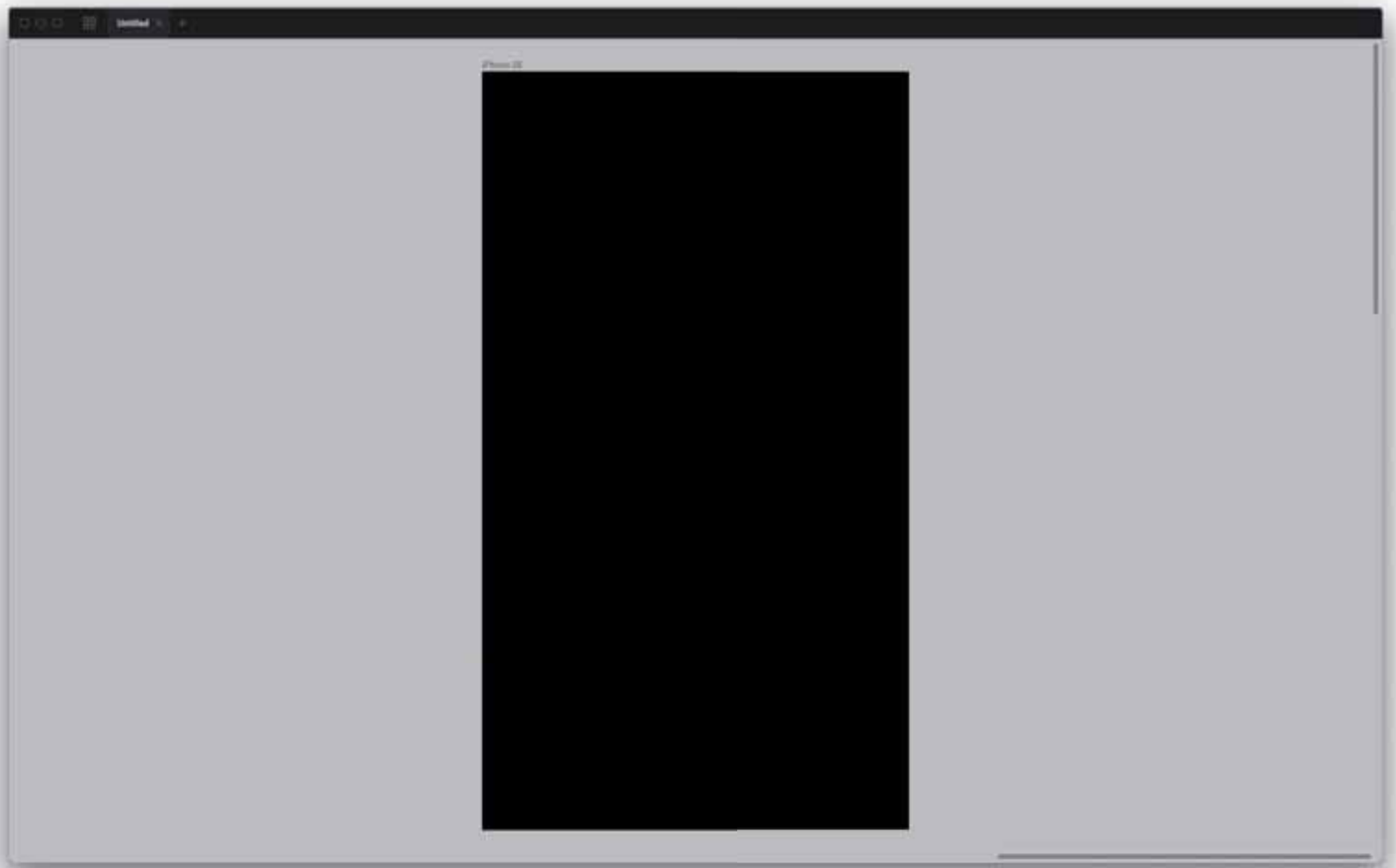
Скрытие интерфейса

Cmd + | скрывает боковые панели.

Не путаем символ **|** [пайп] буквой **I** [ай].

На клавише **|** нарисованы **Ё** и ****.

Боковые панели скрыты:



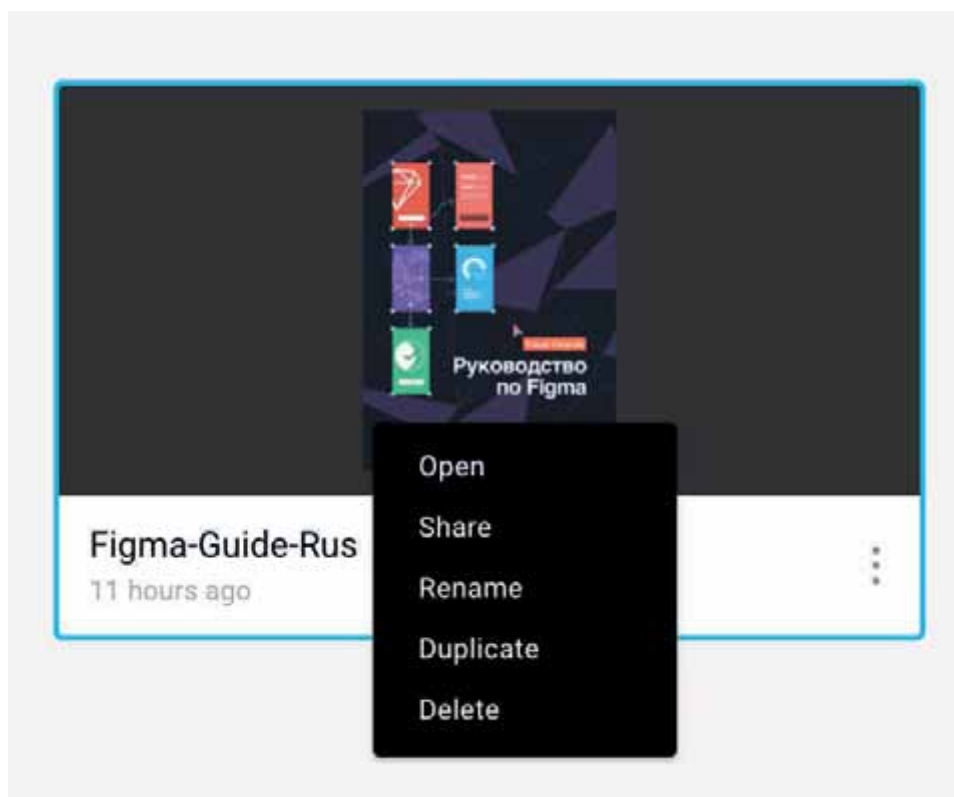
Переименуем файл

В центре экрана указаны **название проекта / имя текущего файла**.

Сейчас файл называется **Untitled**. Если кликнем по названию, мы сможем редактировать его.

Также переименовывать файл можно, находясь на экране проекта.

В контекстном меню есть пункт **Rename**:

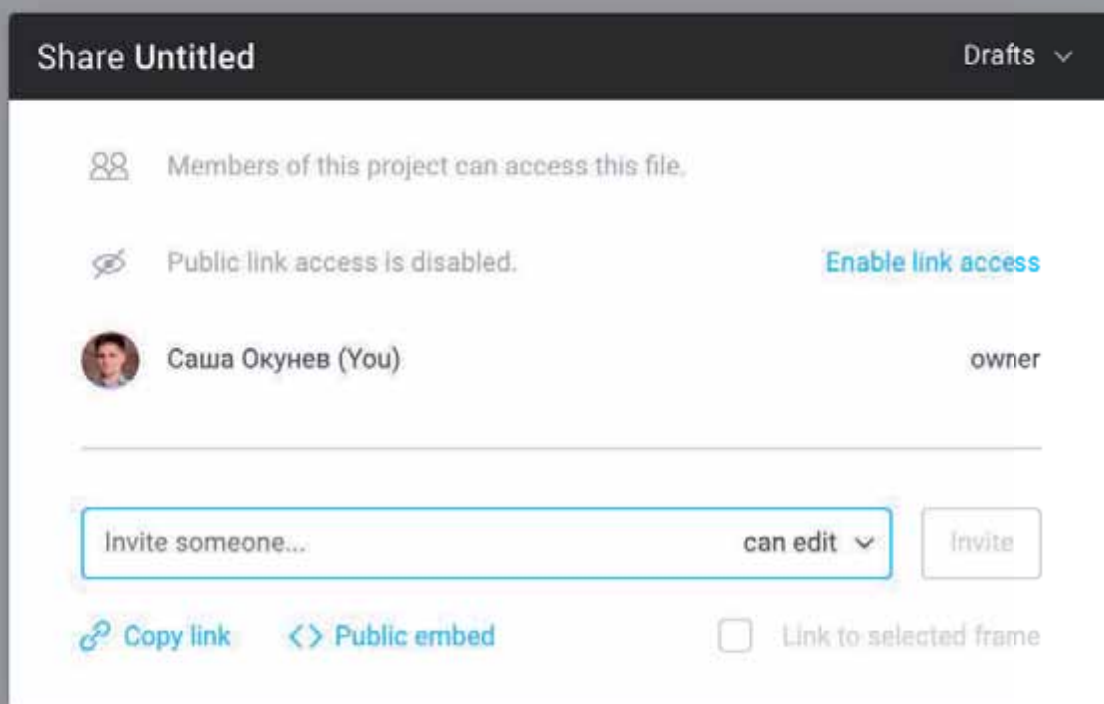


Как давать доступ по ссылке

Голубая кнопка **Share** в правой части верхней панели позволяет настроить, кто будет видеть проект.

Кнопка **Enable Link Access** позволяет создать публичную ссылку, которую могут видеть даже те, у кого нет аккаунта на Фигме.

Давая доступ по ссылке, нужно помнить, что тот у кого есть такая ссылка имеет возможность сохранить исходник проекта себе. Это может быть нежелательно, для тех дизайнеров, которые переживают, что их работу украдут, не заплатив. Фигма работает над решением этой уязвимости.



Практика: тулбар и ссылки на проект

1. Разобраться, как переключать инструменты и возвращаться в **Move**.
2. Сделать публичную ссылку на файл.
3. Открыть её в браузере, в котором выполнен вход в Фигму.
4. По ссылке в браузере найти курсор, который оставляет десктопный клиент. На нём должно быть твоё имя. Убедиться, что получилось синхронно смотреть файл с двух разных окон.

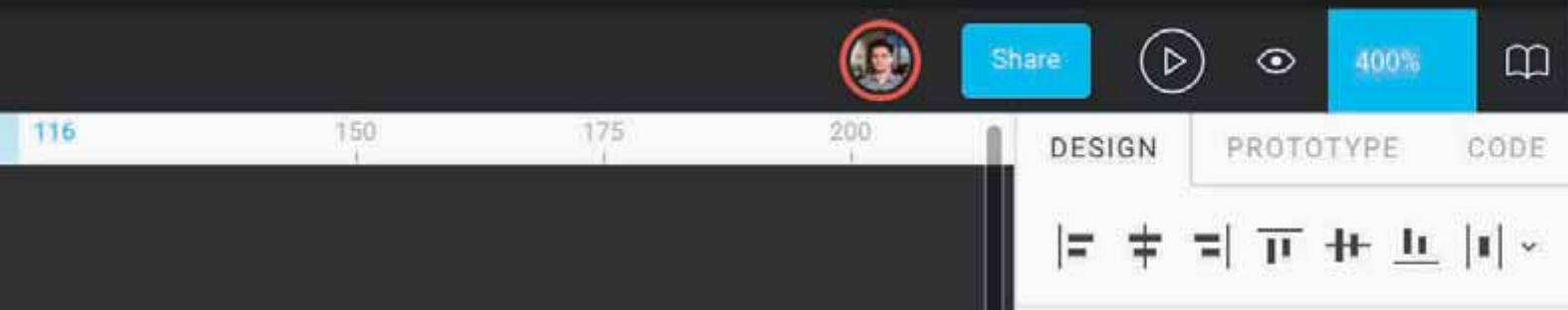
4. По центру: рабочая область

В центре экрана мы видим серый безразмерный холст. Это наша вселенная, в которой мы можем творить. Холст может быть огромен и его достаточно для создания проекта любого размера, хотя на самом деле он не бесконечный.

По холсту можно ползать, зажав **пробел** и перетаскивая пространство левой клавишей. На Маках для навигации по холсту удобно использовать трекпад. На Windows вместо пробела также можно зажимать **среднюю кнопку мыши**.

Существуют нулевые координаты: **x: 0, y: 0**, это центр вселенной, относительно которого позиционируются все слои, у которых нет родительских слоёв.

Слой любого типа имеет базовые свойства: координаты по **x, y, ширину, высоту** и **угол наклона**.



Работаем с масштабом

В окошке **Zoom** в правом верхнем углу отображается значение зума в процентах. Можно кликнуть его и ввести нужное.

Plus Приближает вдвое

Minus Отдаляет вдвое

Shift + 1 Zoom To Fit, аналог **Cmd + 1** в Скетче.

Выставляет такое значение зума и кадрирование, чтобы весь контент страницы влез в рабочую область. Удобная команда, чтобы охватить взором всю страницу.

Shift + 2 Zoom To Selection, аналог **Cmd + 2** в Скетче.

Выставляет такой зум и кадрирование, чтобы выделенный объект занял максимум пространства рабочей области. Команда, которая экономит много сил на настройку масштаба окна под определённый слой на холсте. Выделена ли маленькая иконка или группа экранов, эта команда покажет выделенное во весь доступный размер рабочей области.

Shift + 0 Zoom to 100%. Фактический проектировочный масштаб.

Для гиков

Слои могут находиться в диапазоне от -66 400 px до +66 000 px. Они могут иметь и ещё более экстремальные значения, но перестаёт работать **Zoom to selection**, **Shift + 2**. Это делает такой дальний космос бесполезным.



Фон рабочей области

Можно задать его цвет во вкладке **Design**.

Стоит учитывать, что он должен быть таким, чтобы мелкие надписи в заголовках экранов читались. Иногда они пишутся голубым и фиолетовым цветом.

Глаз напротив строки с цветом позволяет делать прозрачный фон рабочей области с каноническими шашечками.

В качестве тёмного фонового удобен серый #333333, а в качестве светлого — #F2F2F2. Чтобы установить эти цвета, достаточно ввести в поле **Background** с клавиатуры **33** или **F2**.

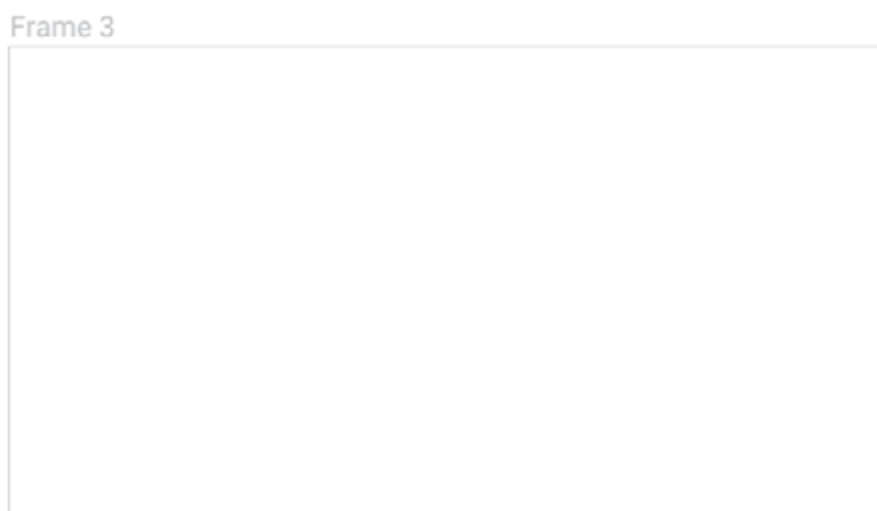
Делать фон тёмным удобно когда долго долго работаешь над проектом, особенно в тёмной комнате. Чем меньше света попадает в глаза, тем комфортнее.

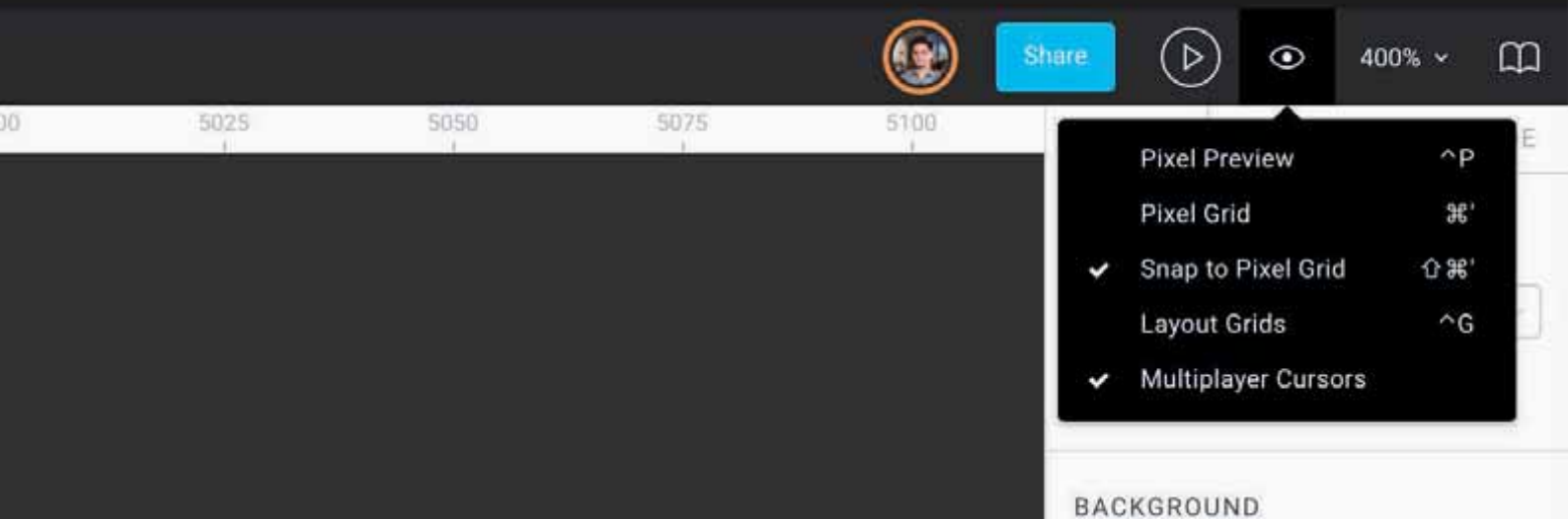
Настройка цвета в определённой странице глобальна для всех участников команды, поэтому следует договориться о том цвете, который комфортен для всех.

Не стоит делать холст белым, иначе фреймы будут недостаточно контрастны, даже если их контуры видны.

Чтобы проявлять и скрывать контуры фреймов (экранов дизайна), используем команду **Frame Outlines**, **Cmd + /**, затем продолжаем печатать: **ou**.

Недостаточно контрастный фрейм будет трудно найти на холсте, а фон экрана в нём может слиться с фоном глобального холста:





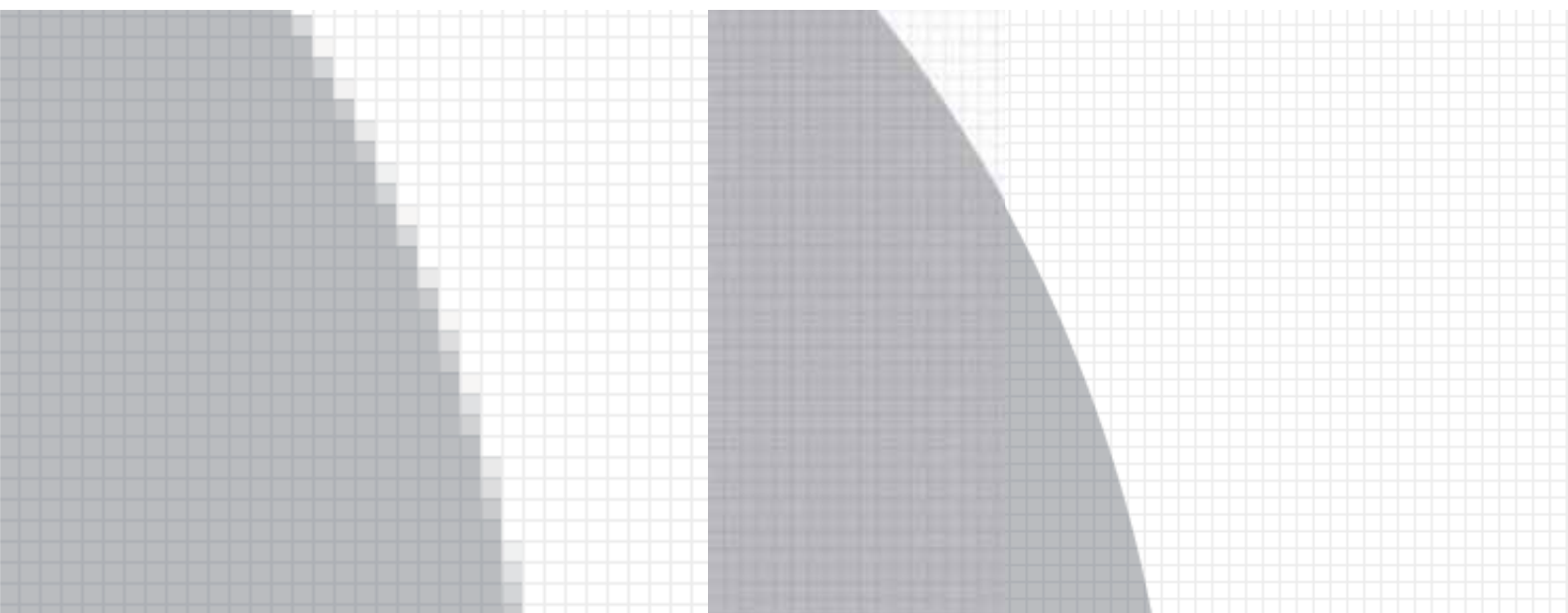
Режимы отображения рабочей области

Иконка глаза скрывает меню режимов отображения. В нем можно настроить, что видно в рабочей области.

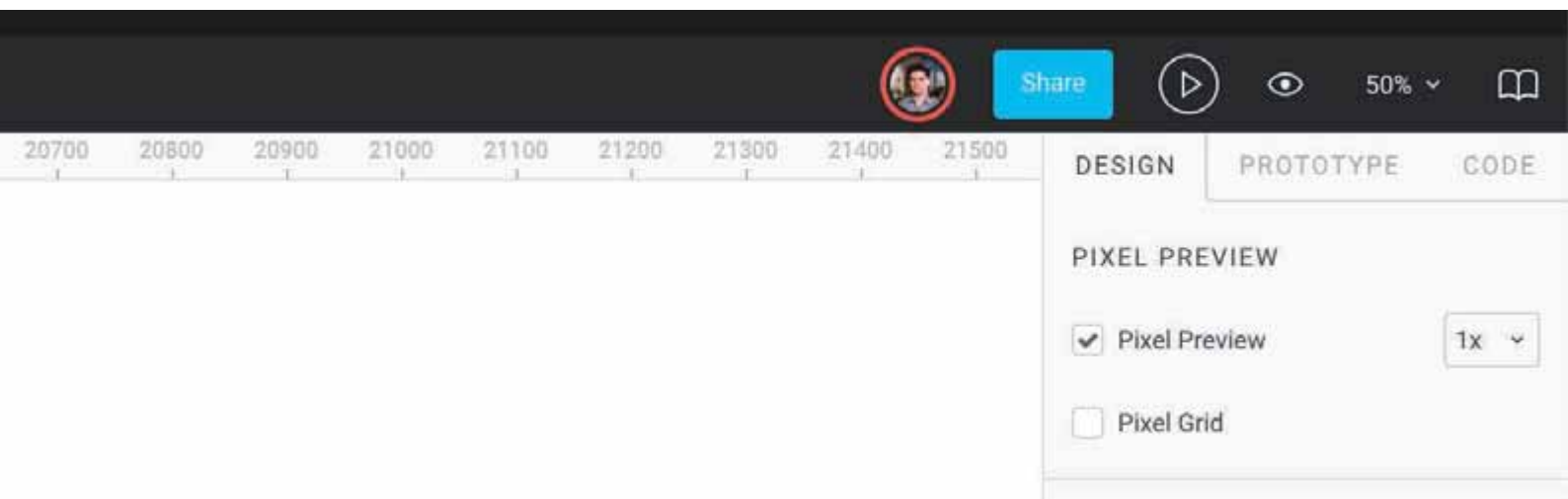
Ctrl + P **Pixel Preview** — в котором видны пиксели в плотности 1x или любом другом, настроенном в выпадающем поле.

На Windows для этой команды используется клавиша **Ctrl + Alt + Y**.

Слева: включен, справа: выключен.



Pixel Grid, **Cmd + '** — тогл-режим¹, в котором показывается пиксельная сетка. **'** на той же клавише, что и **Э**.



Также режимы **Pixel Preview** и **Pixel Grid** доступны во вкладке **Design**.

¹ Тогл-режим — режим работы, который может быть либо включен, либо выключен.

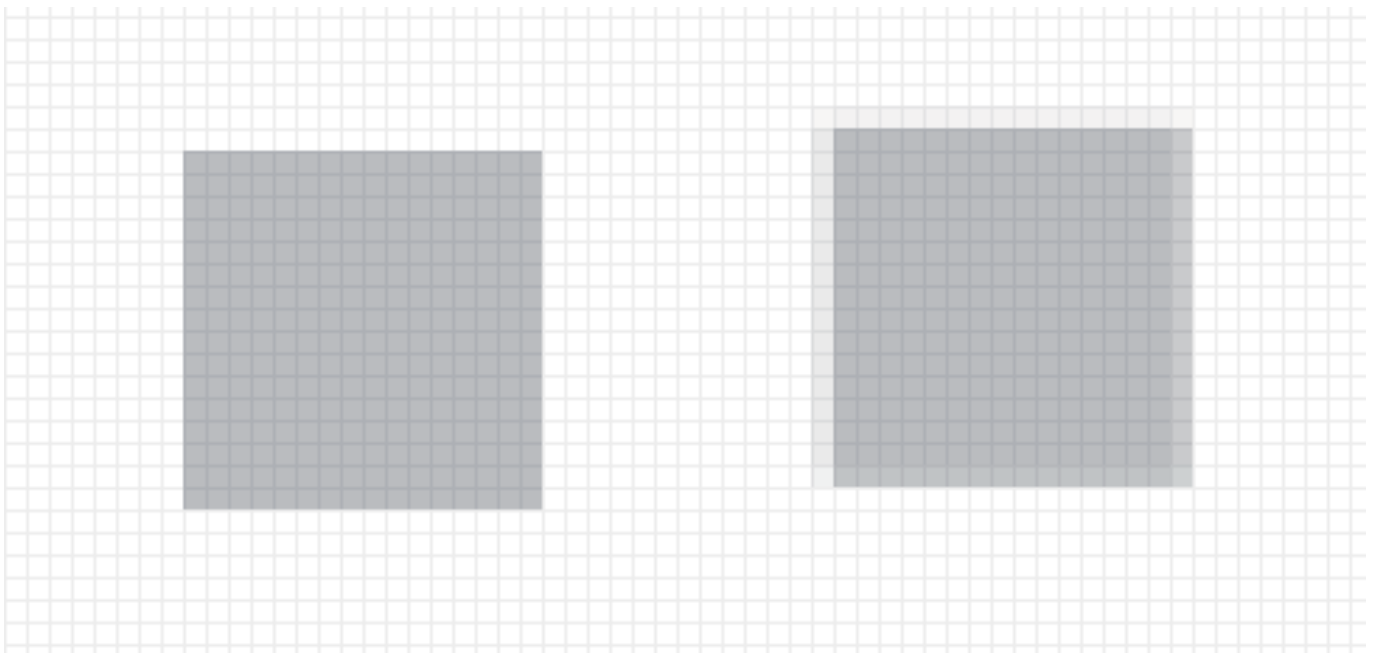
Snap To Pixel Grid, **Shift + Cmd + '**

— тогл-режим привязки слоёв к сетке во время создания или перемещения. Если активен, все новые или движимые слои ставятся на ровные пиксели. Если отключен, они могут попасть на дробные значения пикселей (полупиксели). В этом случае координаты будут иметь числа после точки: 2.29. При плотности пикселей 1x, контуры такой фигуры, будут отображаться нечётко.

Чтобы создать на холсте серый квадрат, нажмём **R** и кликнем в любом месте.

Ctrl + G **Layout Grids**

— тогл-режим, который включает и отключает сетки для всех фреймов. Разберём сетки в главе о фреймах.

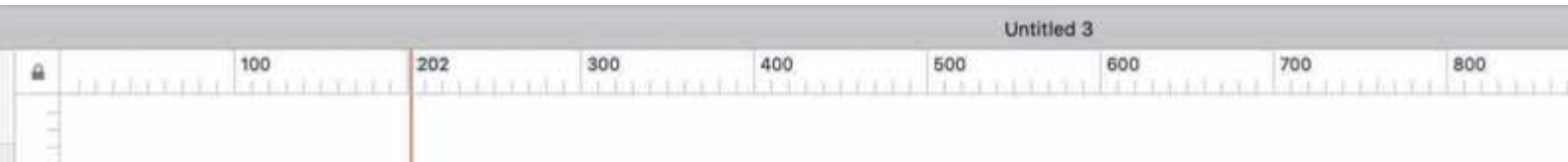


Чтобы не отвлекаться на курсоры других пользователей, которые одновременно работают в файле, отключим их:

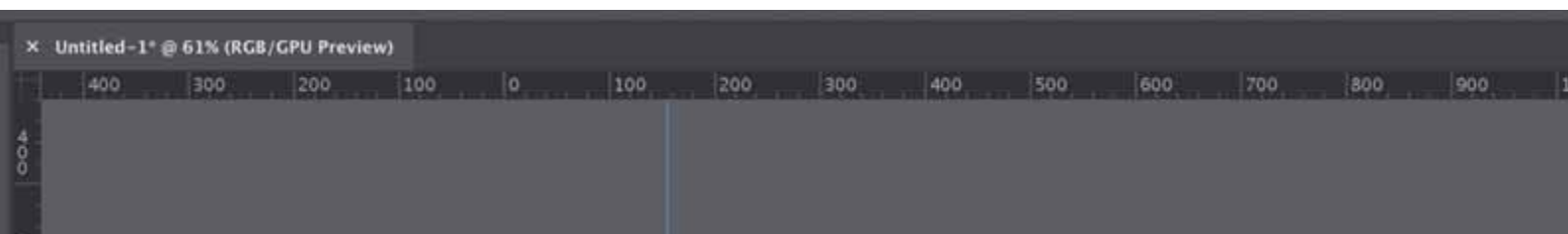
View settings → Multiplayer Cursors

Линейка

Чтобы включить линейку в Скетче, используем **Ctrl + R**, команда **Show / Hide Rulers**. Направляющие ставятся кликом на перпендикулярной оси, в данном случае, **X**:



В Иллюстраторе и фотошопе команда называется так же и имеет клавишу **Cmd + R**. Направляющие нужно вытягивать из параллельной оси, в данном случае, **Y**.



В Фигме команда называется **Rules** и имеет клавишу **Shift + R**. Направляющие вытягиваются так же, как в Иллюстраторе, вытягиванием.



Практика: рабочая область

1. Освоиться с навигацией по холсту: научимся приближать-удалять, переключаться на общий и близкий вид.
2. Создать фрейм.
3. В нём создадим серый квадрат. Приблизим его до крупного масштаба.
4. Включим **Snap To Pixel** и **Pixel Preview**.
5. Передвинем квадрат и почувствуем, как он липнет.
6. Отключим **Snap to Pixel**.
7. Передвинем квадрат и увидим, как поплыли его края.
8. Снова включим привязку. Передвинем квадрат, увидим, что он снова стоит по пиксельной сетке.
9. Включим линейку, если её нет. Скроем если есть.
10. Поставим гайд (направляющую) по левому краю нашего квадрата.
11. Дублируем квадрат, зажав **Opt** / **Alt** и оттащив получившийся второй квадрат от первого. Поставим его ниже первого, выравнивая по гайду.

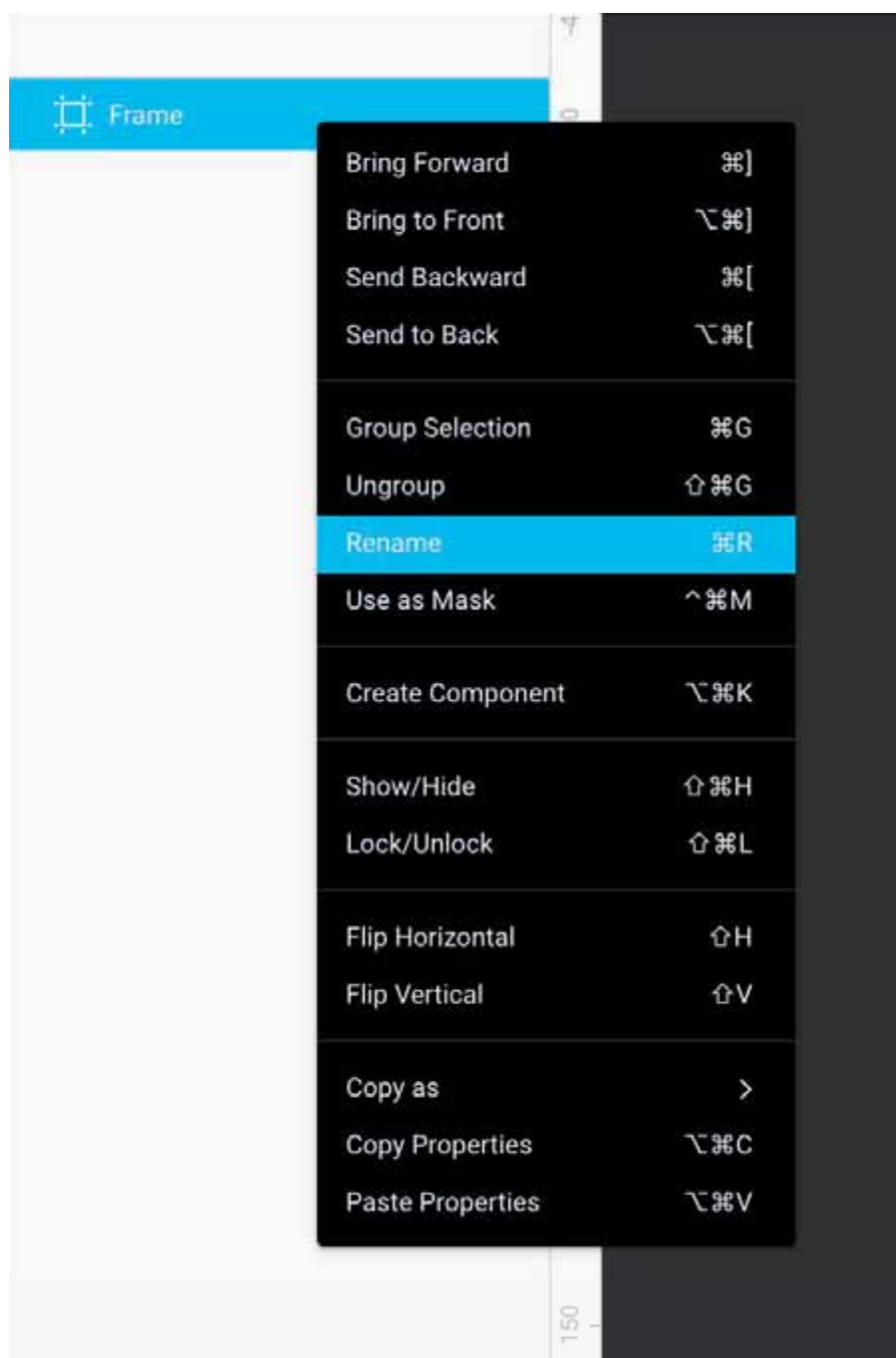
5. Слева: панель слоёв и страницы

Панель слоёв состоит из двух вкладок:

Layers, Opt + 1

Components, Opt + 2

В первой вкладке мы видим список всех слоёв, которые существуют в рабочей области. У каждого слоя есть контекстное меню, где можно посмотреть большинство горячих клавиш:



Полезные клавиши для работы со слоями

Cmd + D Дублировать слой. Новый появится в списке слоёв ниже первого. На холсте он будет правее своего родителя.

Если нажимать несколько раз, слои будут дублироваться на том же отступе, который был между оригиналом и первым дублем.

Также очень мощная техника — зажать **Opt** и тянуть слой, который мы хотим дублировать.

Cmd + R Переименовать слой, **Enter** для сохранения имени

Tab Выделить следующий

Shift + Tab Выделить предыдущий

Enter Развернуть текущий фрейм, компонент или группу

Shift + Enter Выделить родительский фрейм

В режиме переименования можно менять фокус на соседние слои при помощи **Tab** и **Shift + Tab**. Слой, с которого мы убрали фокус, сохраняется.

Shift + Cmd + L Заблокировать или разблокировать слой.

Заблокированный остаётся видимым, но его нельзя двигать и выделять.

Shift + Cmd + H Скрыть / показать слой. В отличие от Скетча, в Фигме можно скрывать экраны.

Сортировка

Cmd +] или **Opt + Cmd + ↑** Перетащить слой на одну позицию вверх

Cmd + [или **Opt + Cmd + ↓** Вниз.

Opt + Cmd +] Перетащить слой в самый верх списка.

Opt + Cmd + [Поставить последним

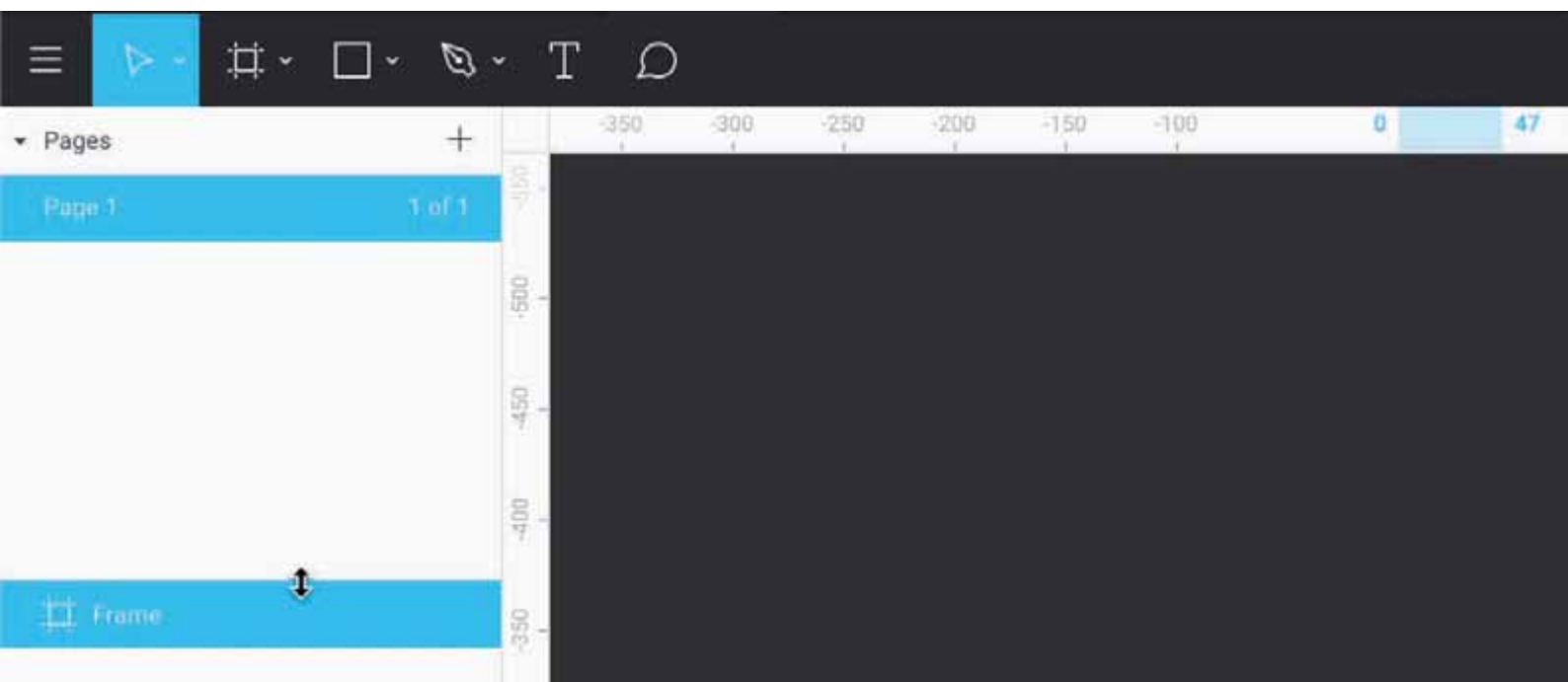
Страницы

Как и в Скетче, страницы в Фигме — секции, в которых содержатся наши экраны. Удобны для группировки их по смыслу. В каждой странице своя бесконечная координатная плоскость. Есть где развернуться!



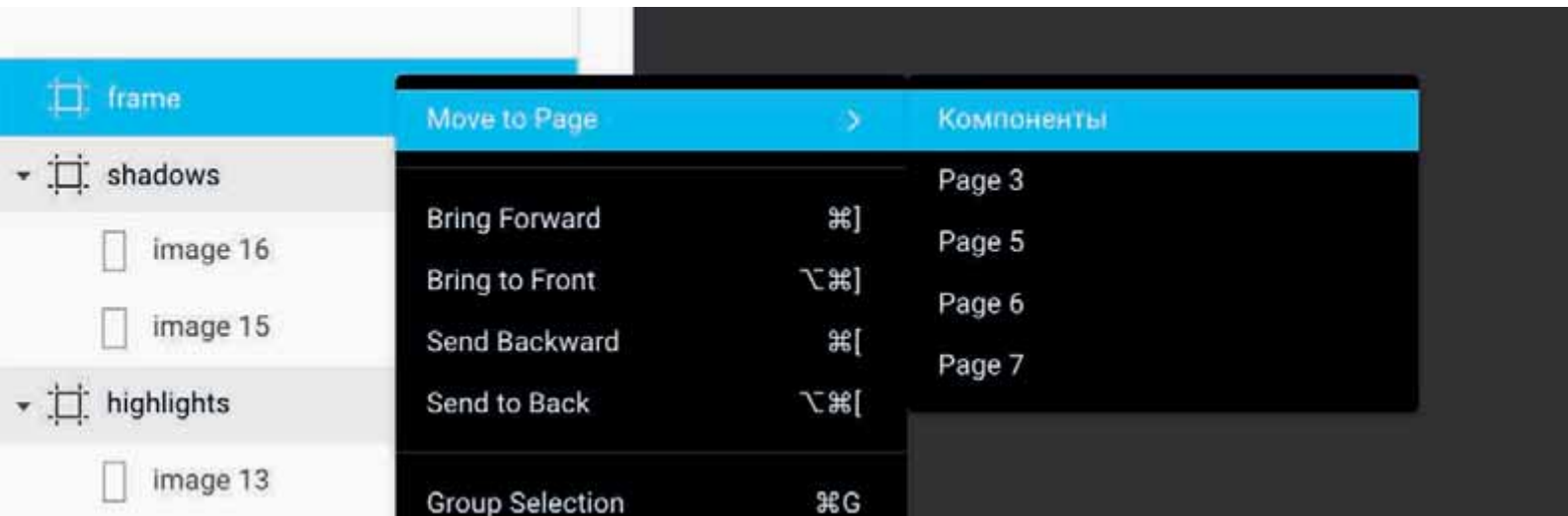
fn + ↑↓ Листаем страницы с клавиатуры. Windows — **PageUp**, **PageDn**.

Клик на строку **Page 1** разворачивает список страниц. Кнопка **+** создаёт страницы. Если в файле слишком много страниц, можно растянуть вниз и дать им больше пространства:



Перемещение слоёв между страницами

В отличие от Скетча, в Фигме не работает перетаскивание фрейма в другую страницу. Для перемещения следует использовать команду **Move To Page** в контекстном меню.



Практика: список слоёв и страницы

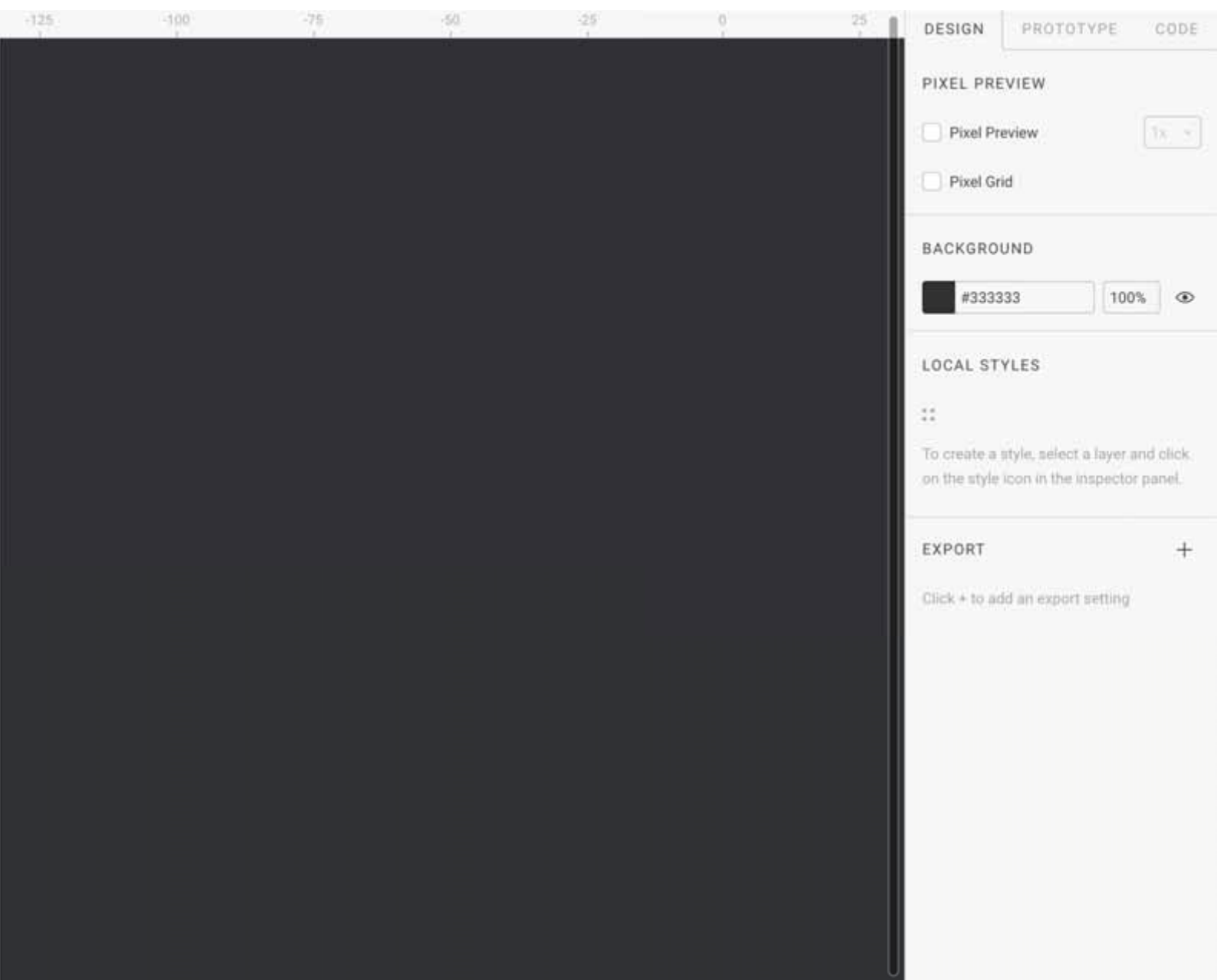
1. Создадим фрейм.
2. Дублируем его 3 раза, чтобы получилось 4 фрейма.
3. Пронумеруем их от верхнего до нижнего: 1 → 4.
4. Поставим фрейм 1 нижним.
5. Второй заблокируем.
6. Третий скроем.
7. Создадим страницу Page 2. Выясним, как переименовывать страницы.
8. Отправить фрейм 4 на неё.

6. Справа: панель свойств

В правой части интерфейса видна большая белая панель во всю высоту окна — панель свойств. В зависимости от выделенного типа слоя в ней может быть разный набор полей и кнопок.

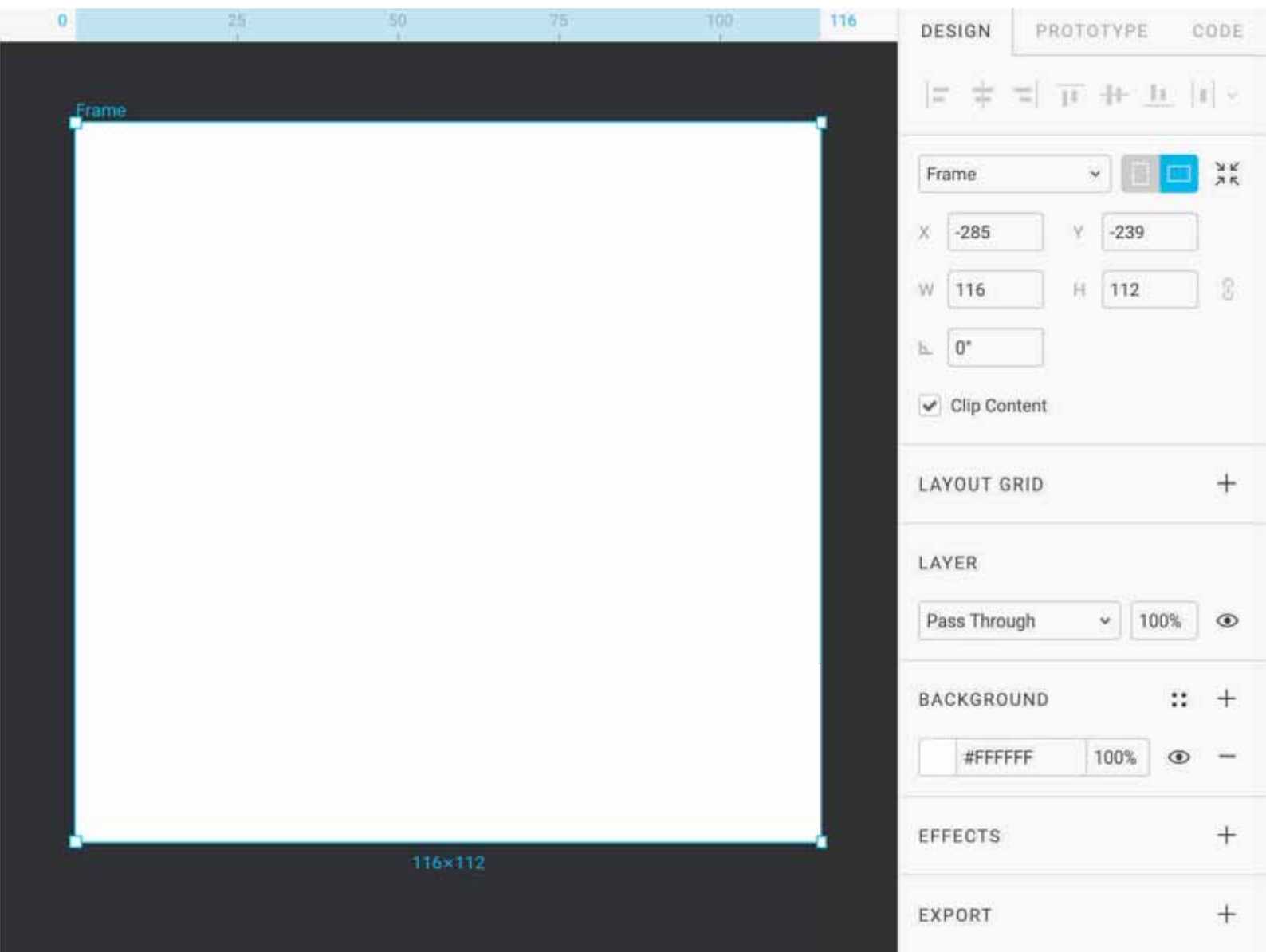
Если ни один слой не выбран, мы видим настройки файла.

В такой панели видны уже рассмотренные нами режимы просмотра **Pixel Preview**, настройка цвета рабочей области, изначально пустой блок **Local Styles** для палитры, а также блок **Export** для массового экспорта.



В панели свойств есть три вкладки: **Design**, **Prototype** и **Code**. Для создания дизайна мы работаем с первой.

для примера создадим слой типа фрейм, **F**. Если выделить его, панель справа отобразит его свойства:

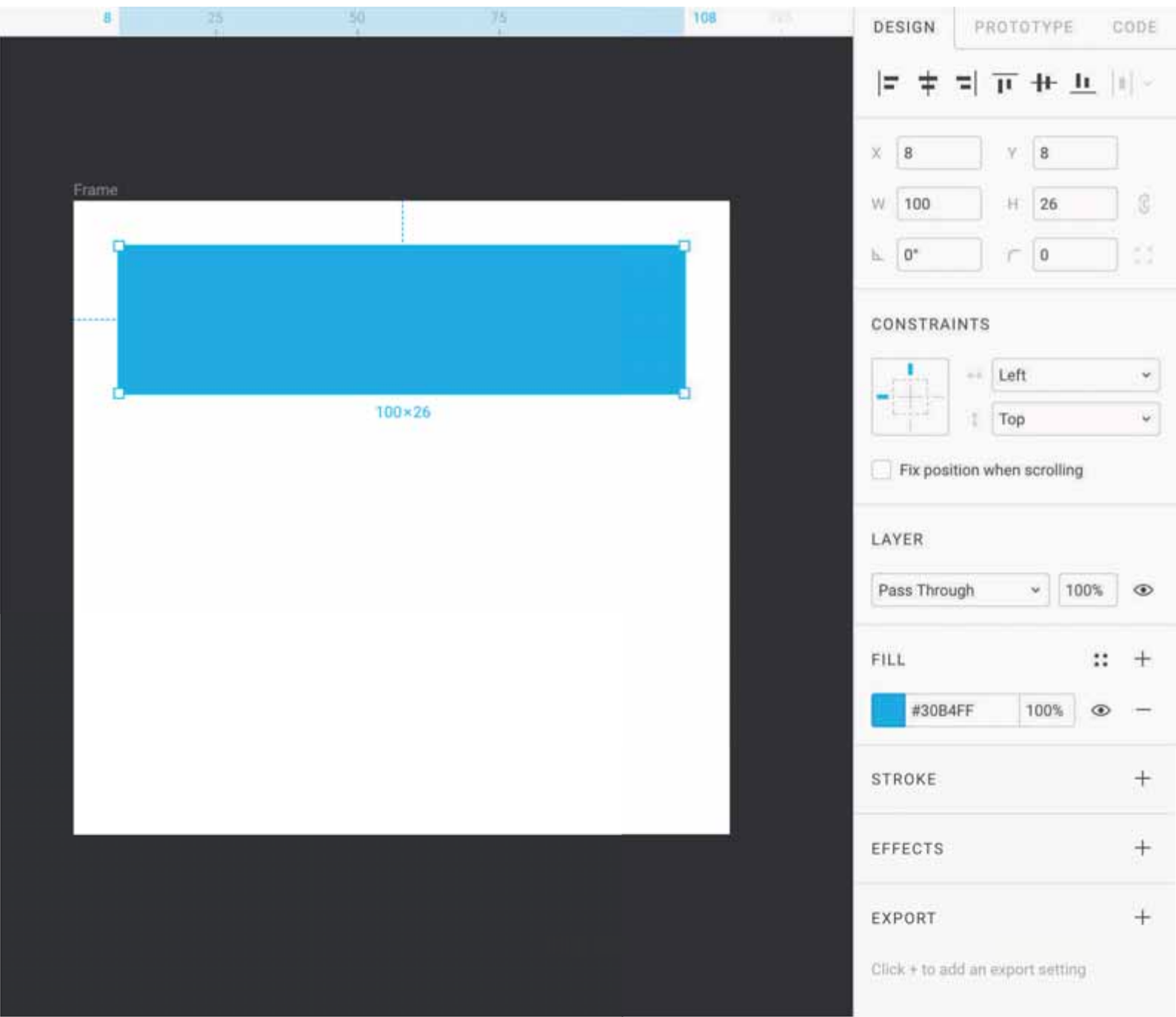


Сверху мы видим настройки выравнивания, изначально недоступные, потому что в пустом фрейме нечего выравнивать. Можно настроить размер фрейма под размер экрана определённого устройства. Для этого выберем нужное из длинного списка доступных устройств.

В блоке **Layout Grid** настраивается сетка, в **Layer** — режим наложения, в **Background** — фон, в **Effects** тени и блюр. В блоке **Export** можно настроить, в какие форматы слой будет экспортироваться.

В фрейме создадим прямоугольник, **R**.

Если выделена векторная фигура (шейп), боковая панель имеет характерные элементы для этого типа слоёв: например, поле скругления углов, которого не было у фреймов. Блок **Constraints** для определения, как будет вести себя шейп, если будем растягивать его родительский фрейм. Подробнее об этой теме поговорим в главе **Адаптивность и ограничители**. Блок **Stroke** служит для работы с обводкой.



Общие свойства слоёв

Слой любого типа имеет базовые свойства:

- **X, Y** — координаты в пикселях. Если слой на бесконечном холсте, абсолютные. Если внутри фрейма — относительные.
- **Width, Height** — ширина и высота в пикселях
- Правее ширины — кнопка **Constrain Proportions**, позволяет сохранить пропорции ширины и высоты, если они меняются в одном из полей.
- **Angle** — угол наклона. Любой слой можно наклонять на этот угол.
- **Opacity** — опасность². Любой слой, включая фреймы, может быть полупрозрачным. Можно задавать с клавиатуры: **1** — 10%. **5** — 50%. **0** — 100%. Если нажать две клавиши быстро, введём точное значение: **0**, **5** — 5%. **5, 5** — 55%.
- **Layer mode** — режим наложения. По умолчанию **Pass Through**.

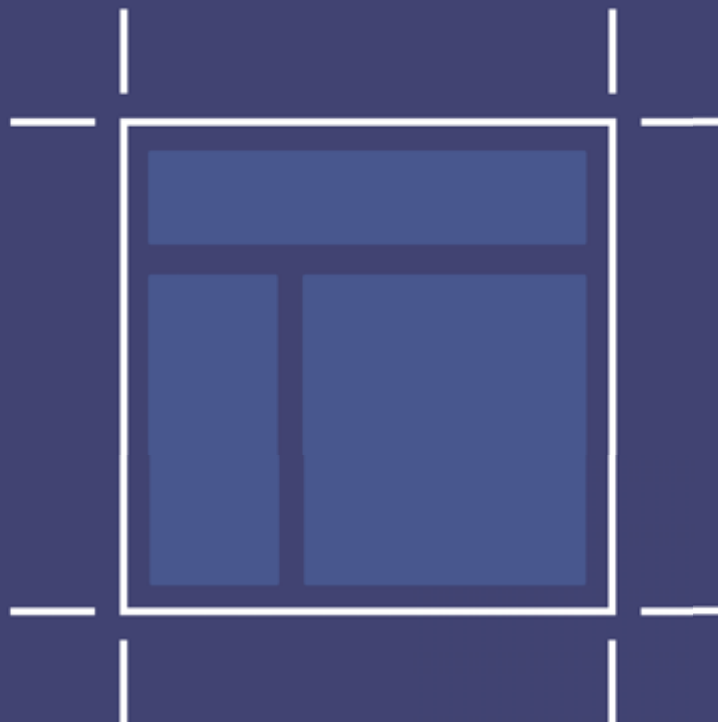
У всех векторных типов: шейпов, текстовых слоёв и векторных сетей есть такие свойства как обводка **Stroke** и заливка **Fill**.

Corner Radius — радиус скругления углов у шейпов.

² Опасность — удачный сленговый термин для кривого слова «непрозрачность», который предложил использовать Илья Бирман из Бюро Горбунова.

Практика по свойствам слоя

1. Создадим фрейм.
2. Зададим ему ширину 1600, высоту 900.
3. Зададим ему координаты: 0, 0.
4. Пусть фон его будет чёрный.
5. Внутри него создадим белый квадрат размером 100 x 100 и радиусом скругления 10.
6. Выровняем квадрат по центру фрейма.
7. С клавиатуры зададим опасность квадрата в 20%.
8. Экспортируем фрейм в PNG-файл.



7. Фреймы

[Проект в Фигме →](#)

О том, как работать с панелью слоёв начали говорить в главе 5. Рассмотрели также, что каждый тип слоёв имеет свои уникальные настройки в панели справа. Более близкое знакомство со слоями начнём с фреймов.

Фреймы — аналог артбордов в Скетче. Как мы узнали в прошлой главе, создаются клавишей **F** в режиме инструмента **Frame**.

Фреймы очень важны при работе с сеткой. Им можно задавать единые стили сеток, которые можно использовать по всему файлу.

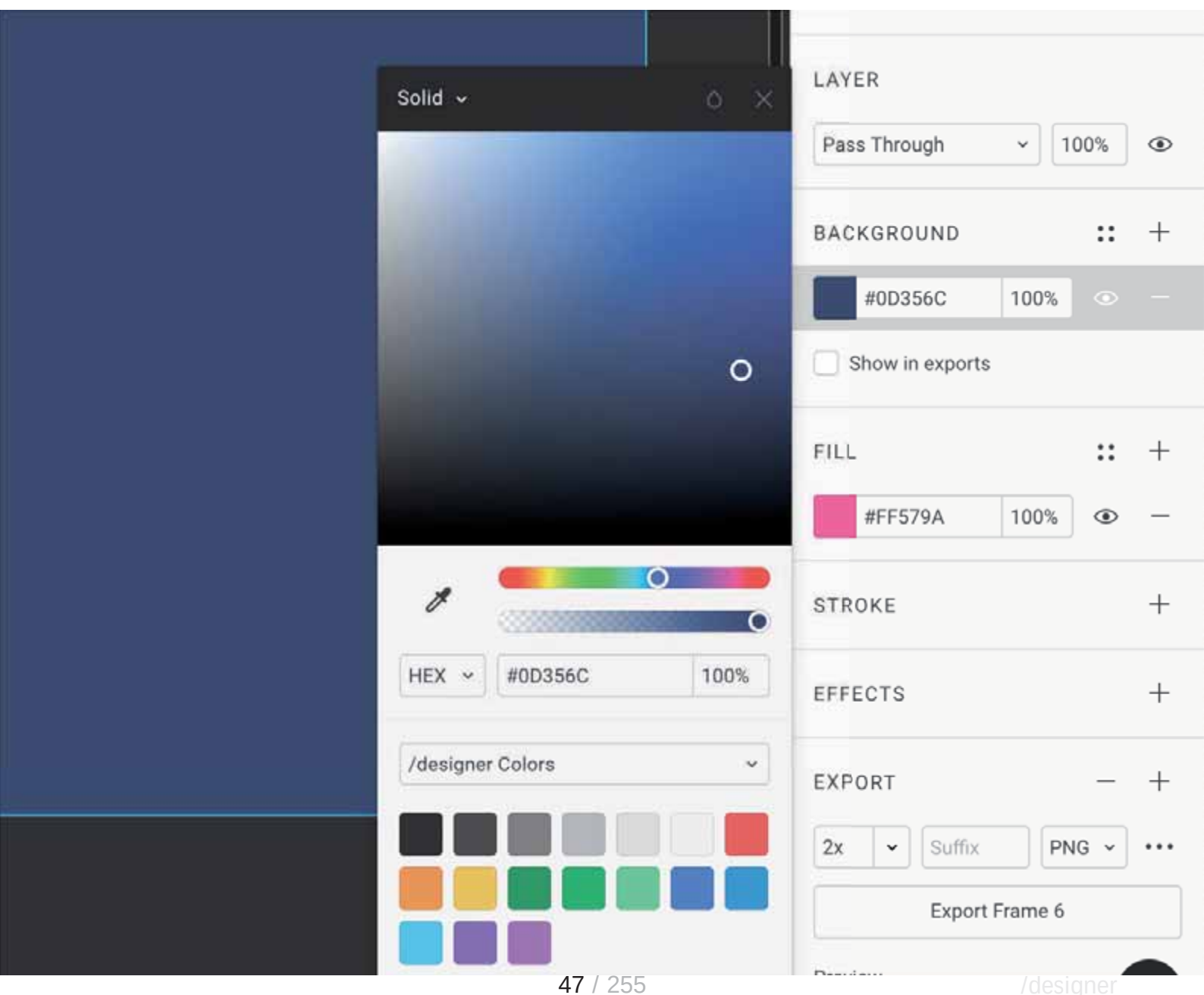
Имеют фоновый цвет — **Background**, изначально белый. Галочка **Show in exports** позволяет отображать фоновый цвет при экспорте фрейма в файл. Если её снять, фрейм будет выгружен в PNG или иной формат с прозрачным фоном.

Галочка **Clip Content** позволяет выбрать, будет ли виден слой, части которого вылезают за пределы фрейма. В Скетче всегда режется.

Фоновый цвет фрейма

Фреймы могут иметь фоновый цвет, который задаётся в блоке **Background**. Изначально заливка фона стоит в режиме **Solid**, что даёт равномерное наложение цвета.

Как и в Скетче, в Фигме действует клавиша **Ctrl + C**, которая вызывает **Color Picker** — пипетку. На Windows используется **I**. Пикер позволяет снять нужный цвет в любом месте проекта.



Фреймы: Работа с сеткой

В фреймах можно настраивать сетку. Для этого нужно выделить фрейм и в блоке **Layout Grid** нажать **+**. Это создаст в фрейме новую сетку.



Ctrl + G показывает или скрывает сетку по всем фреймам в проекте.

Добавили вторую сетку:

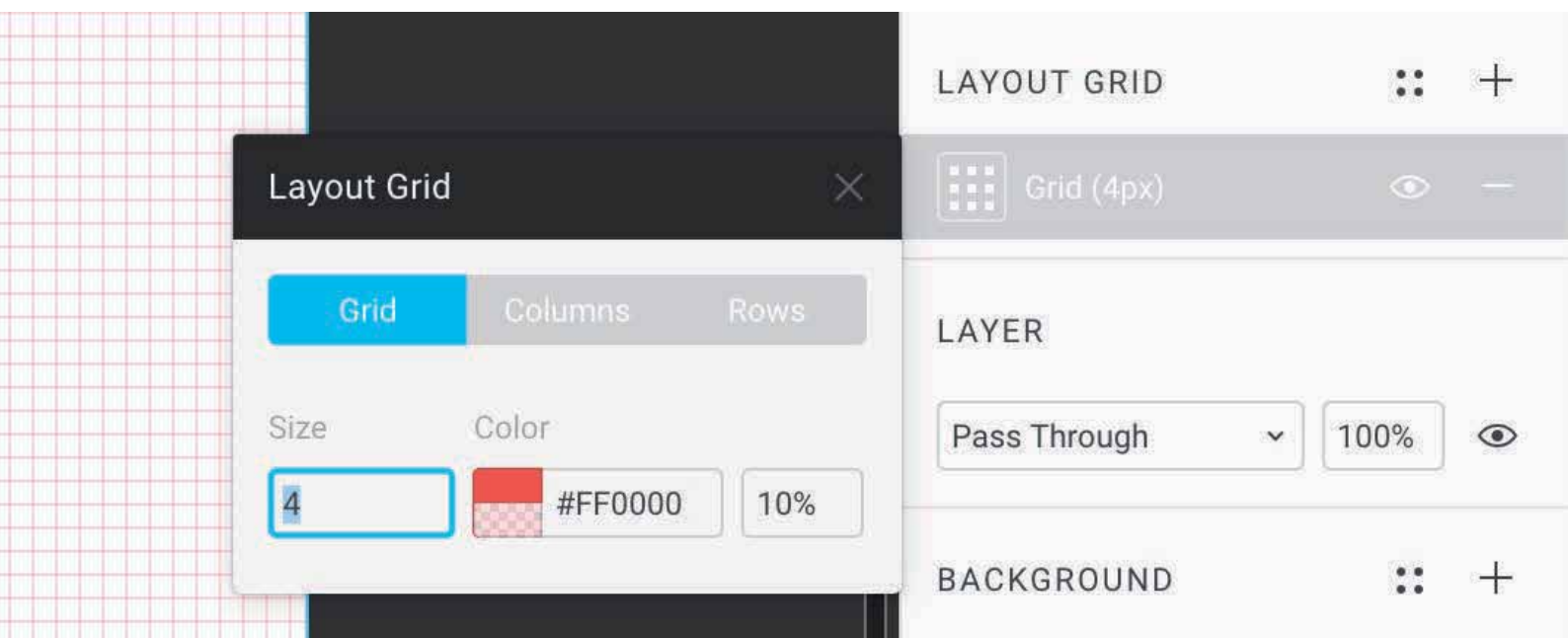


Сетка может быть трёх типов: **Grid**, **Columns**, **Rows**. Они могут накладываться друг на друга и выстраиваться в список в блоке **Layout Grid** подобно слоям. Можно включать нужные сетки, нажав глаз.

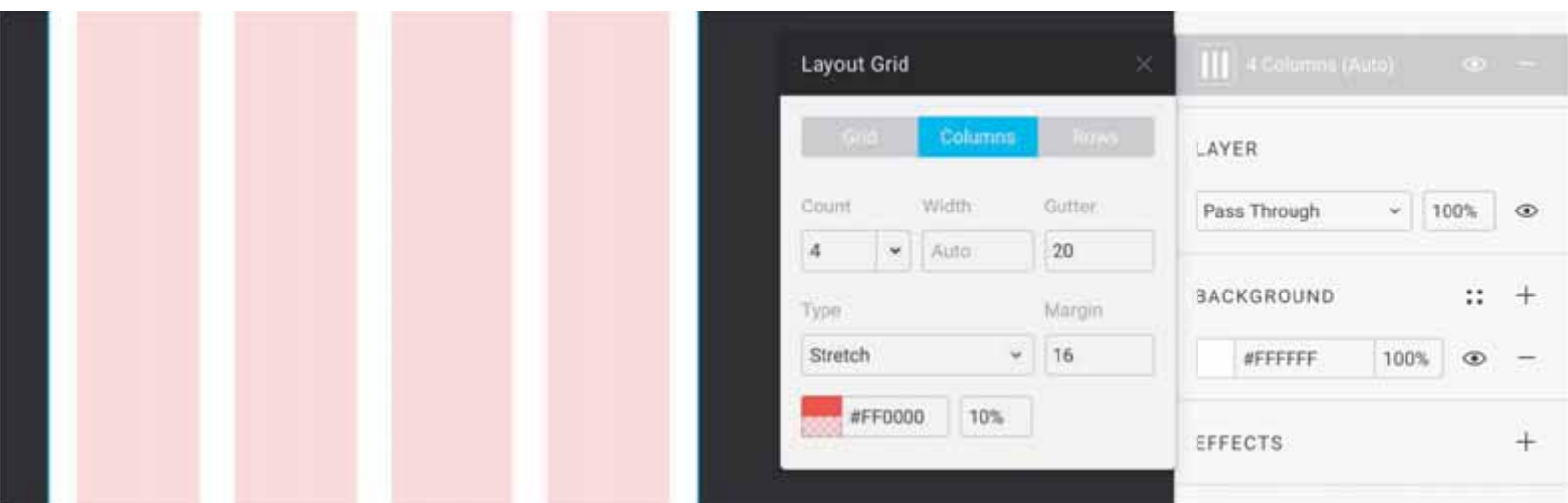
Чтобы настроить тип сетки, нажимаем на иконку:



Grid – базовая регулярная сетка.



Columns – модульная вертикальная сетка.



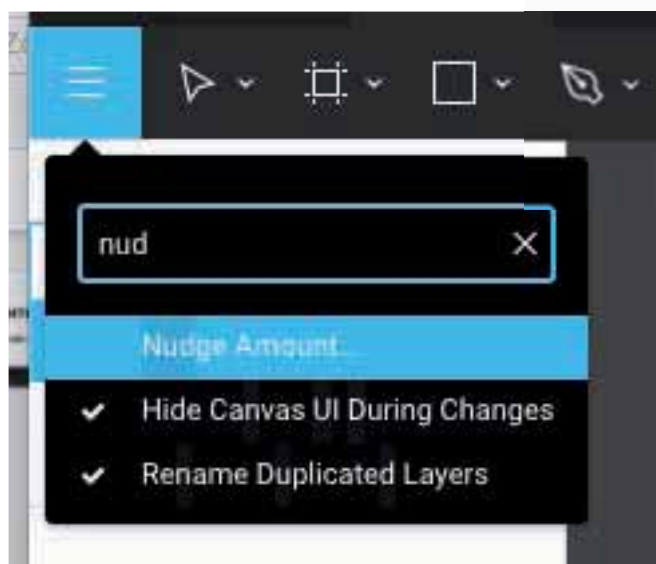
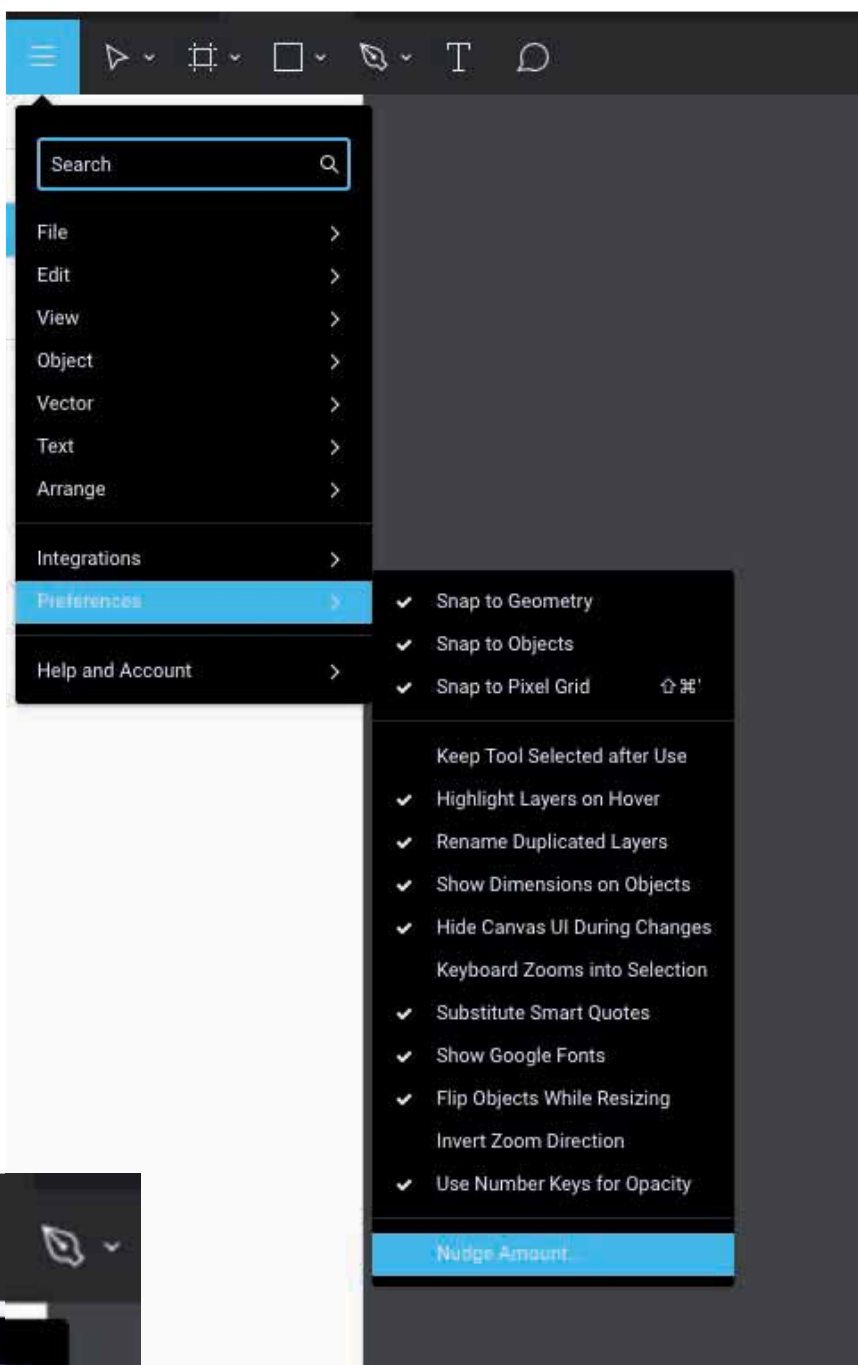
Rows – модульная горизонтальная сетка.

Шаг сдвига по сетке

В Фигме можно настроить, на какое расстояние будет сдвигаться объект, если зажать **Shift + стрелки**.

По умолчанию **10**,
Я рекомендую
использовать **4** или **8** —
значение, равное шагу
базовой сетки.

Через поиск будет
быстрее: **Cmd + /**, **nud**

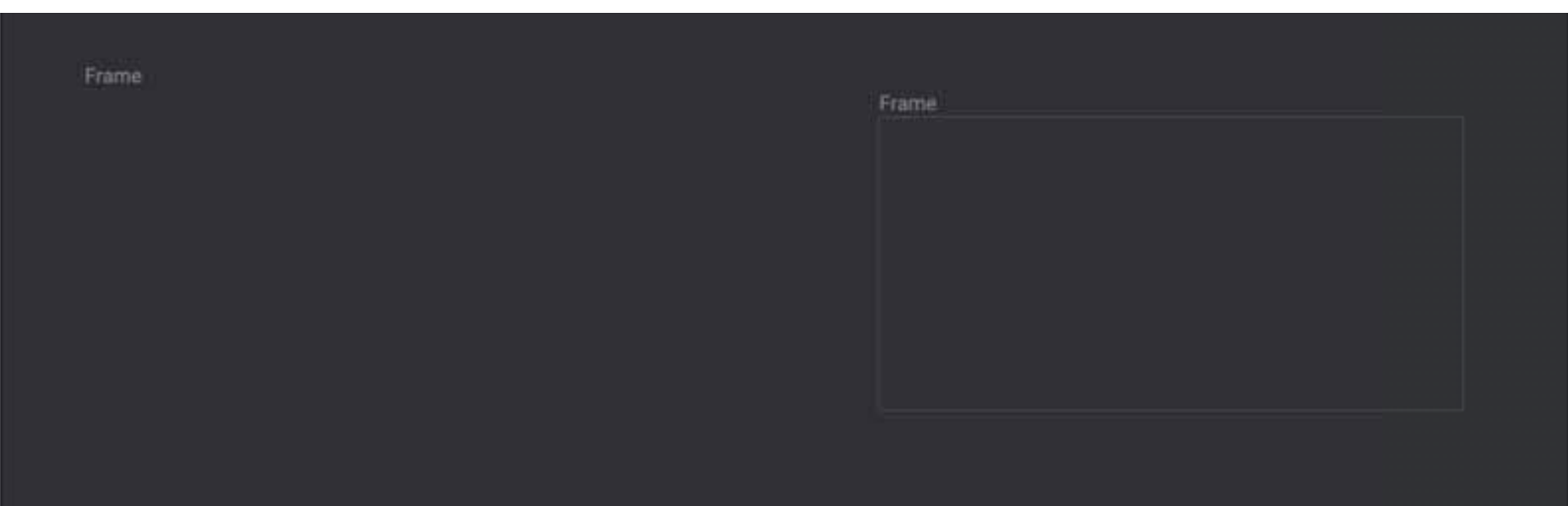


Фреймы для профи

Границы фреймов

Фреймы могут иметь явную границу, которая помогает понимать их размер в ситуации, когда у фрейма нет фона. Для этого используется настройка **View → Frame Outlines**, **Cmd + /, ou**.

Слева **Frame Outlines** выключен, справа включен. Также в определении границ фреймов поможет режим **Outline**, **Cmd + Y**.



Оборачивание фреймом

Любой объект можно за один клик обернуть фреймом. Это бывает иногда нужно, чтобы впоследствии его экспортировать или превратить в компонент.

Для этого используется команда **Frame Selection**, **Opt + Cmd + G**.

Как и артборды в Скетче, фреймы — это разновидность групп. Поэтому, их можно развязывать при помощи **Shift + Cmd + G**.

Если фрейм вокруг слоя уже есть и нужно ужать его до размеров этого слоя, как и в Скетче, используется команда **Resize To Fit**. **Cmd + /, fit**.



8. Шейпы

[Проект в Фигме →](#)

Шейпы (shapes) — векторные фигуры с закрытым периметром. Если периметр разорван, шейп превращается в путь (path).

Фигма позволяет создавать следующие типы шейпов:

R Rectangle

O Ellipse, **O**val

Polygon

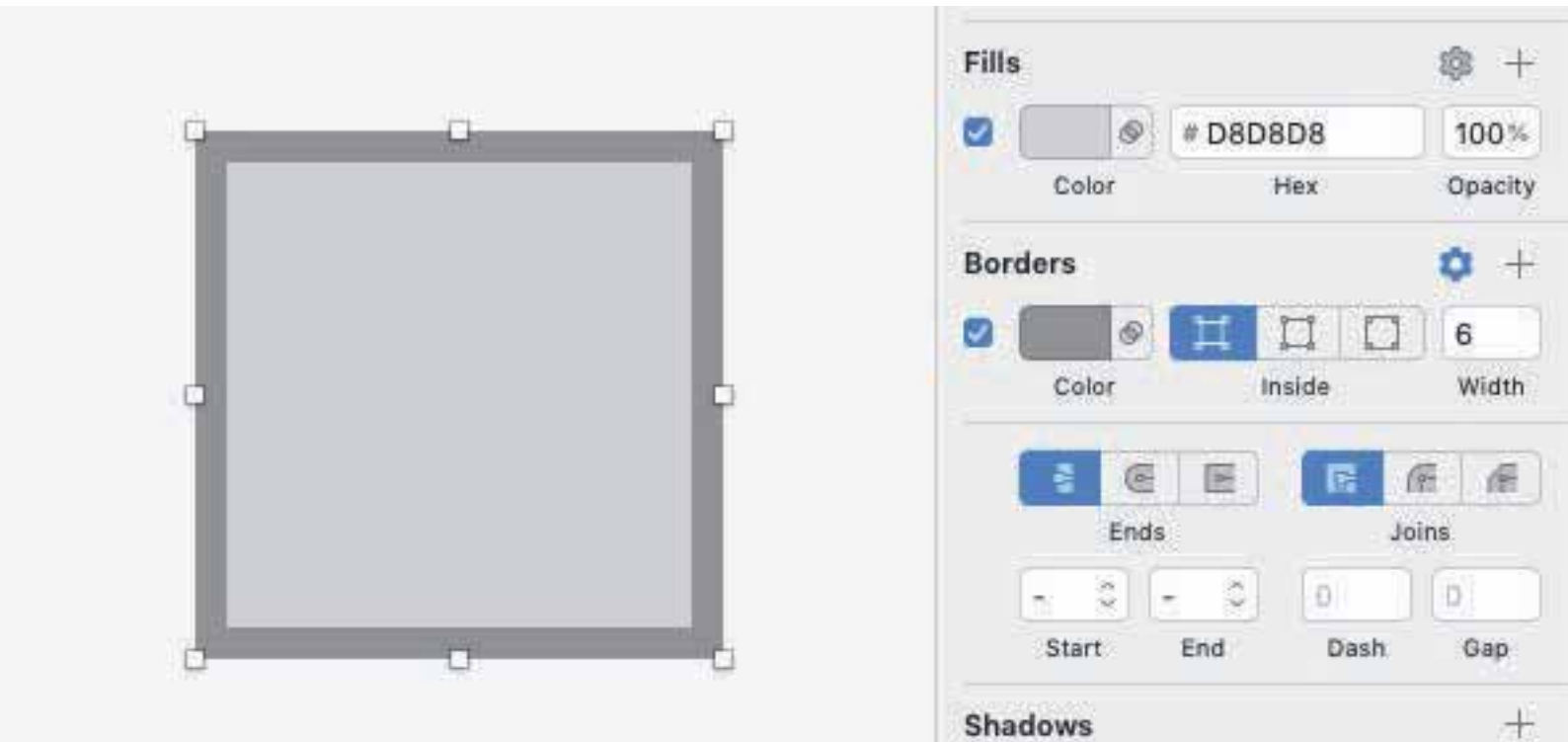
Star

Если выбран инструмент любого шейпа, при клике в рабочую область будет создана фигура размером 100x100.

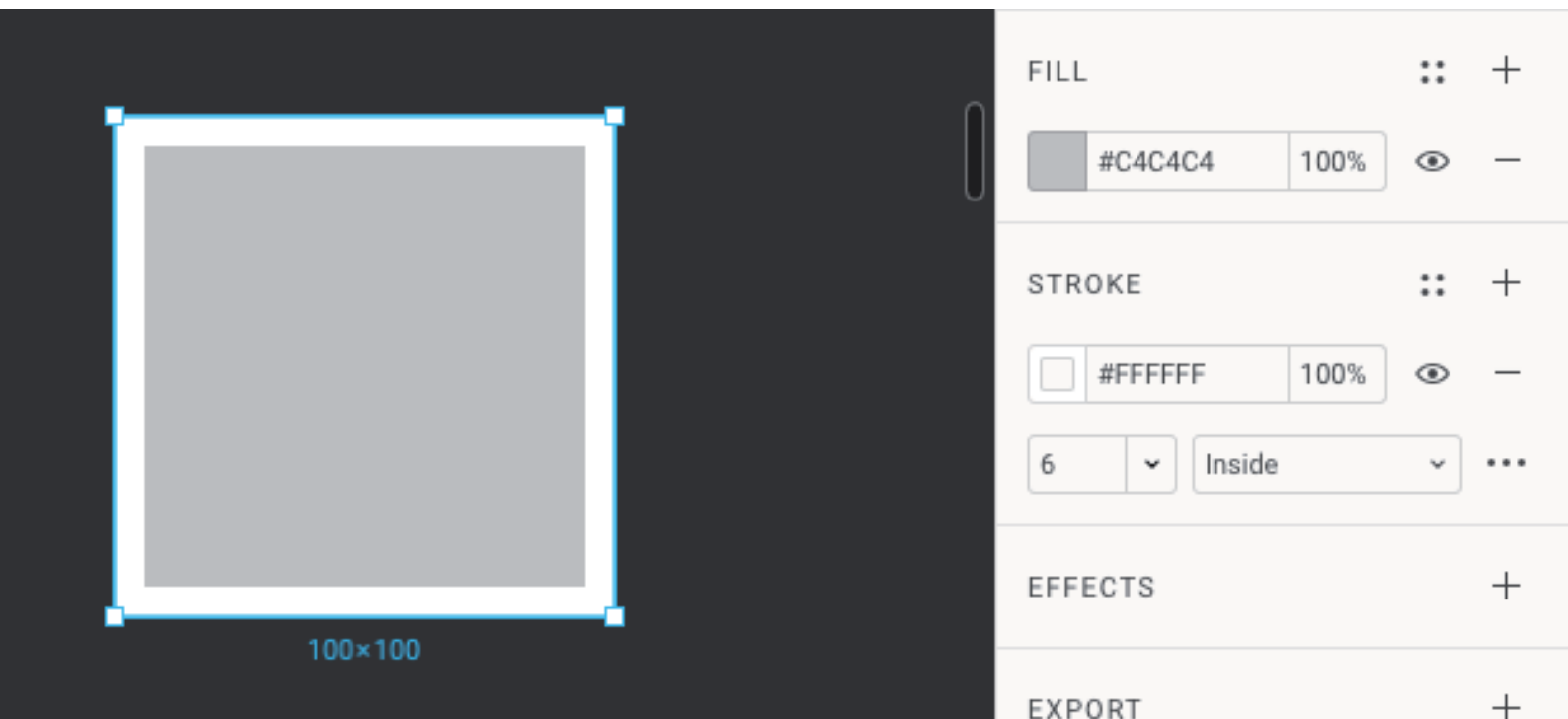
Прямоугольник

Рисуем квадрат: **R, Shift + тянем**. Если зажат **Shift**, во время перетаскивания фигура будет сохранять равную ширину и высоту.

Скетч:



Фигма:



Чем отличаются фреймы и прямоугольники

Цель фреймов в Фигме такая же как у артбордов в Скетче — быть границами макета. Поэтому они заточены под работу с сеткой, имеют предустановленные пропорции популярных устройств и фоновую заливку.

Цель прямоугольных шейпов — быть видимой формой внутри фрейма. Они больше подходят для рисования кнопок и блоков. Поэтому они помимо заливки имеют обводку и скругления углов.

Ещё прямоугольники служат для отображения растровых изображений, которые накладываются на них как заливка.

Линии, которые их формируют, можно разрывать и искажать, уходя от изначальной формы прямоугольника.

Сдвиг и масштабирование

Shift и сдвиг мышью

Фигуру можно перемещать на другое место. Если во время движения зажать **Shift**, мы зафиксируем горизонтальную или вертикальную ось, по которой она сможет перемещаться.

Shift и масштабирование

Фигуру можно тянуть за квадратные ручки. Чтобы сохранять её пропорции во время масштабирования, нужно зажать **Shift**.

Толщина линий при таком масштабировании меняться не будет.

Масштабирование с сохранением толщины линий

Если требуется увеличить или уменьшить объект так, чтобы линии обводки тоже изменились, надо использовать инструмент **Scale, K**.

В отличие от Скетча, в Фигме нет окна, которое позволяет задавать в процентах, насколько точно нужно увеличить объект.



Режим увеличения **Scale** опасен тем, что линии могут получать значения мимо пикселей. Если требуется оптимизировать иконки в 1x, линии мимо пикселей в них будут мыльные.



Заливка и обводка шейпов: первый взгляд

Как и в любом векторном редакторе, в Фигме есть возможность заливать векторные фигуры цветом, а также накладывать по их контуру обводку.

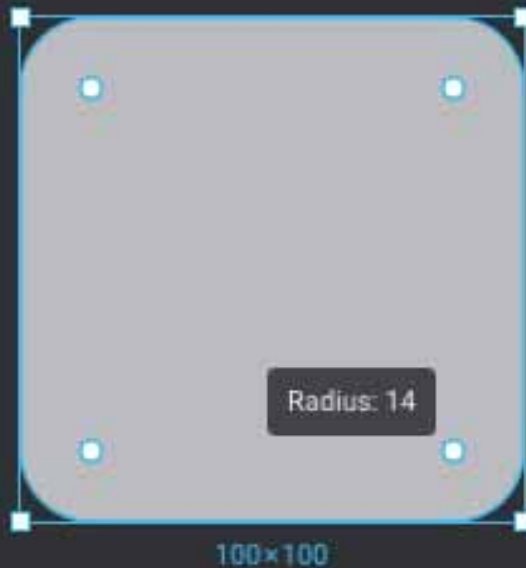
Для работы с заливкой используем блок **Fill**.

За обводку отвечает следующий за ним блок **Stroke**.

В Скетче и Фигме у слоя-шейпа можно настраивать неограниченное количество обводок и заливок, нажимая **+**.

В **Fill** и **Stroke** цвет накладывается на фигуру по одним и тем же принципам, которые мы разберём в отдельной главе **Заливка и градиент**.

Про детальные настройки обводки поговорим в главе **Обводка**.



Скругления

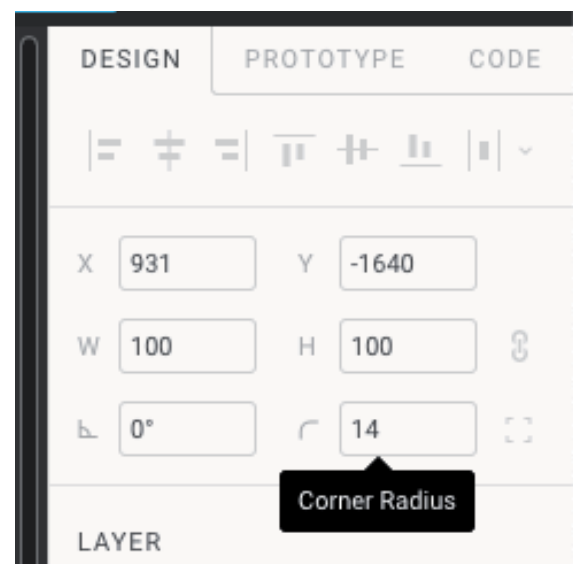
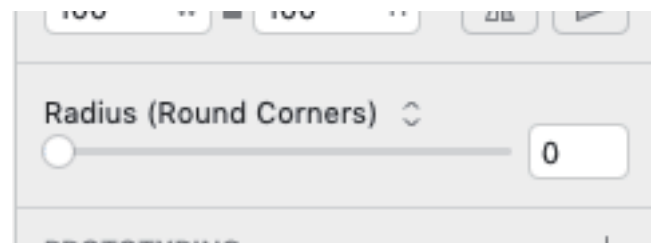
Скругления углов в Фигме и Скетче можно менять.

В Фигме, как и в Иллюстраторе, если навести на выделенный квадрат, внутри него будут видны круглые ручки для управления скруглением. Тянем их к центру, все 4 угла скругляются.

В Скетче для этого приходится использовать фейдер **Radius**.

Эта же настройка будет в блоке координат справа.

Чтобы убрать скругление, выставляем в **Corner Radius** 0 или задвигаем ручки от центра.

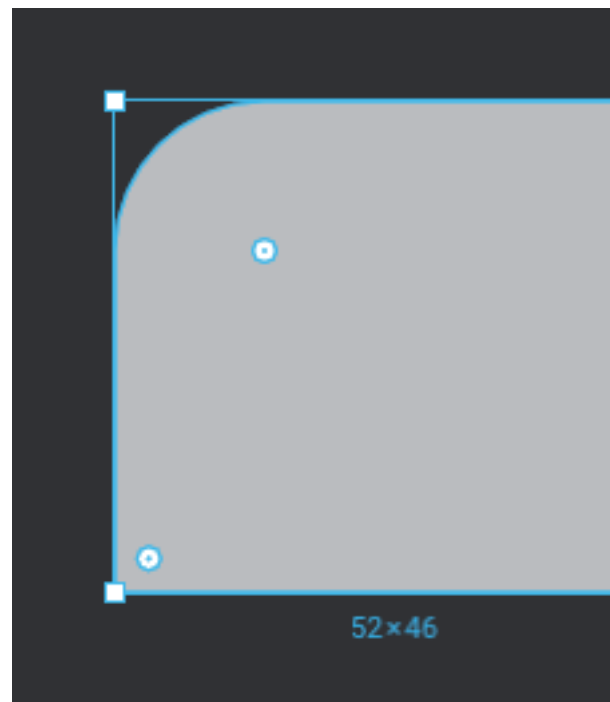


Скругление одного угла

Также можно скруглять каждый угол по отдельности, если зажать **Opt** и тянуть за нужный угол.

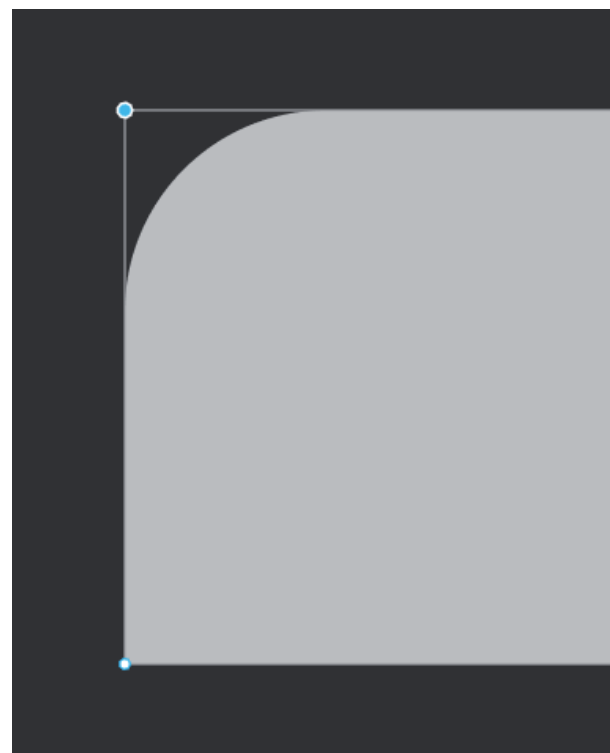
Это рекомендуемый способ скруглять углы. В этом случае, если нужно будет копировать CSS-код, строка **border-radius** попадёт в него:

```
width: 52px;  
height: 46px;  
background: #C4C4C4;  
border-radius: 14px 0px 0px 0px;
```



Второй способ скругления

Через режим редактирования: выделить точку и индивидуально для неё скруглить угол. Способ плох тем, что при копировании CSS Фигма не пропишет строку с **border-radius**. Верстальщики будут не рады. Однако, это неплохой вариант, если наш прямоугольник станет не кнопкой, а частью иллюстрации. CSS-свойства из него никто копировать не будет.



Прямоугольник и сетка

Прямоугольники — мощнейший пластилин для создания блоков, кнопок, отбивок и любых других сущностей, которые должны ложиться в сетку. Если нужна точность, мышью или стилусом с прямоугольниками работать гораздо труднее, поэтому полезно выучить клавиши ниже.

Скетч и Фигма позволяют контролировать прямоугольники и точки в них с клавиатуры с точностью пикселя.

Двигаем фигуру стрелками

Выделяем фигуру, нажимаем **стрелки** — двигаем с шагом 1px.

Shift + стрелки — двигаем на расстояние, настроенное в **Nudge amount**. Как правило, такой шаг равен той сетке, в которой ты работаешь. В нашем случае это 4px.

Shift значительно ускоряет скорость сдвига, позволяя оставаться в принятой сетке.

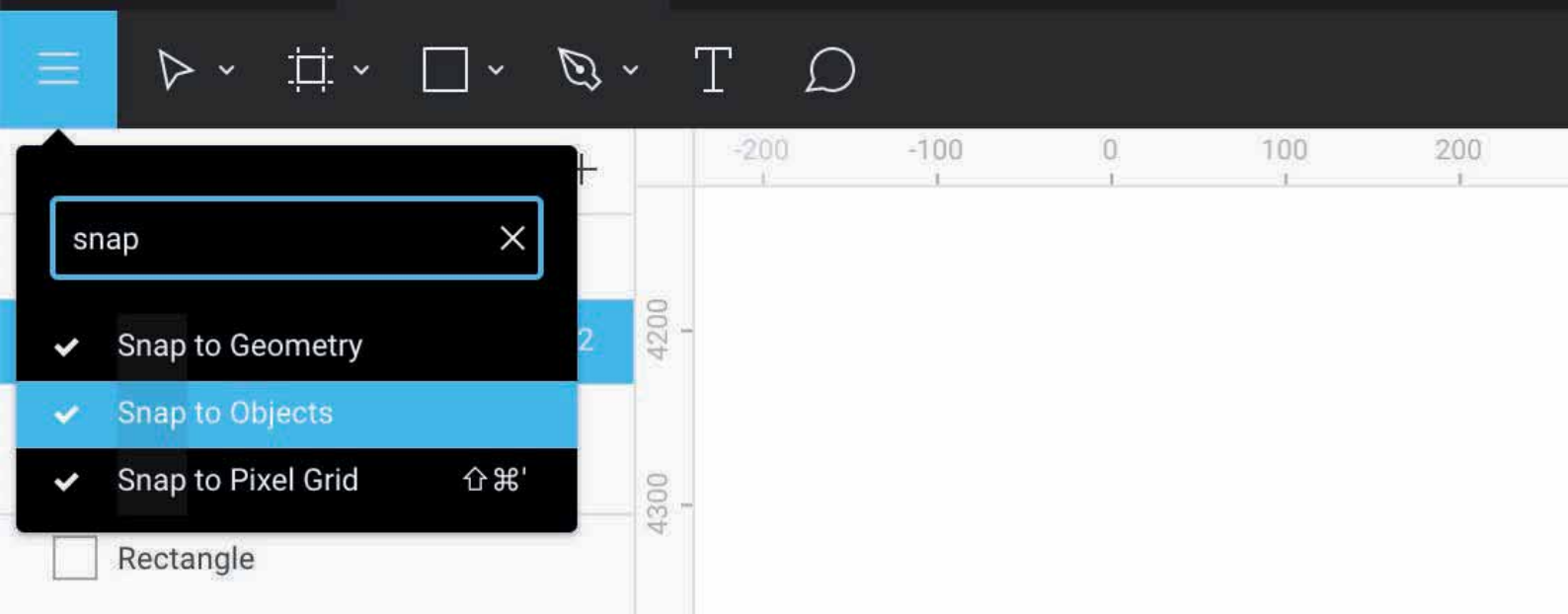
Плющим или тянем фигуру командой

Cmd + стрелки меняют ширину или высоту фигуры на 1px.

Shift + Cmd + стрелки — тянем ширину и высоту на **Nudge amount**. Аналогично сдвигу, не съезжаем с сетки.

Плющим несколько фигур

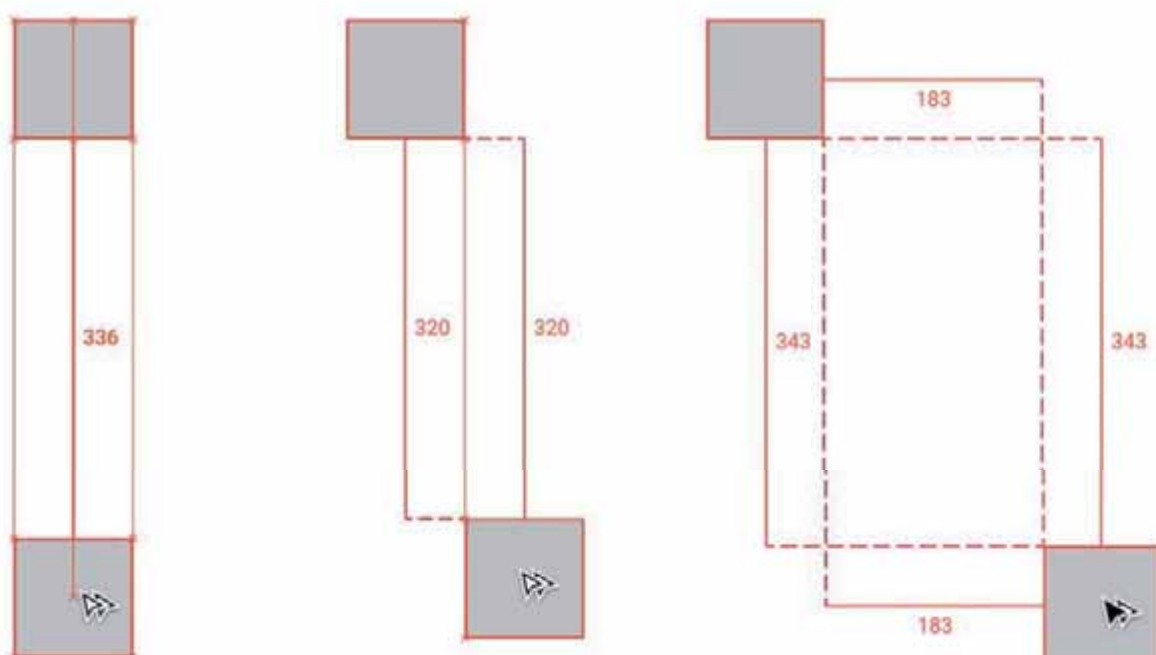
Ещё одно преимущество перед мышью: можно выделить сразу несколько фигур, затем с клавиатуры менять их размер стрелками, а не каждую по очереди мышью.



Сдвигаем и липнем к объектам

Изначально активный режим **Snap to Objects** позволяет двигать фигуры более точно. Во время сдвига фигура магнитится к точкам других фигур. Даже работая мышью, дизайнер может расставлять объекты по холсту с точностью до пикселя.

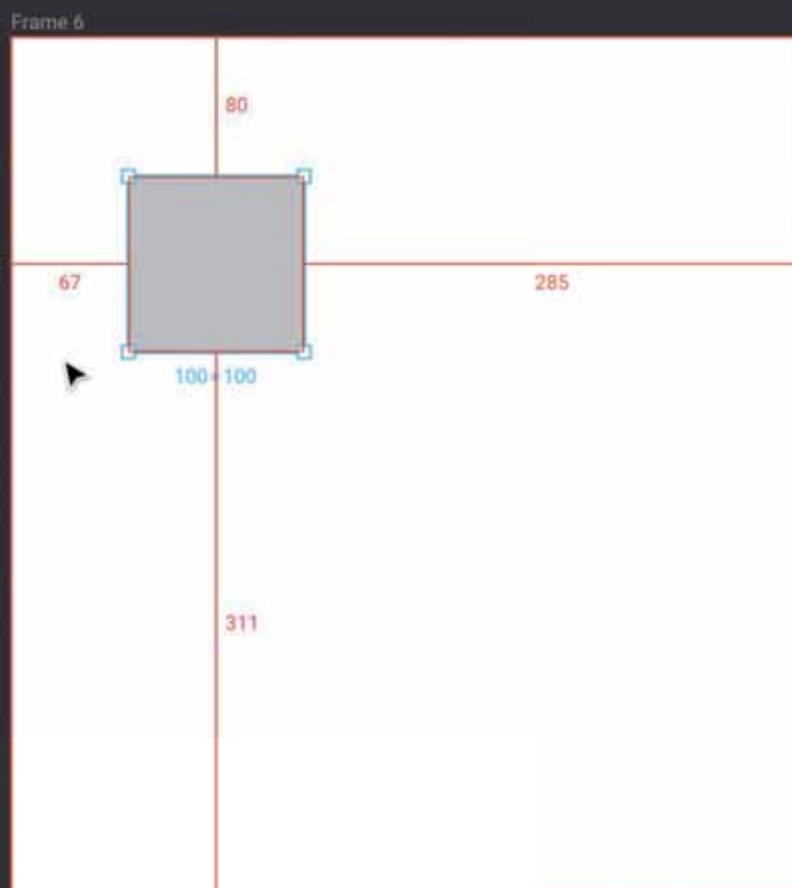
Как в Скетче и Иллюстраторе зажмём **Opt** и потянем квадрат вниз, чтобы дублировать его. Под нижним квадратом будут видны красные линии и расстояния в пикселях, которые подсказывают, как его можно выравнивать по верхнему:



Opt для определения расстояний

Измерение расстояний через **Opt** стало мощнейшей техникой, которая впервые появилась в Скетче. Затем идею подхватили Фигма и Фреймер.

1. Начертим фрейм, **F**.
2. Внутри него кликнем квадрат, **R**.
3. Сместим указатель с квадрата и зажмём **Opt**. Пока указатель находится внутри фрейма, будут видны расстояния в пикселях от границ квадрата до границ фрейма:

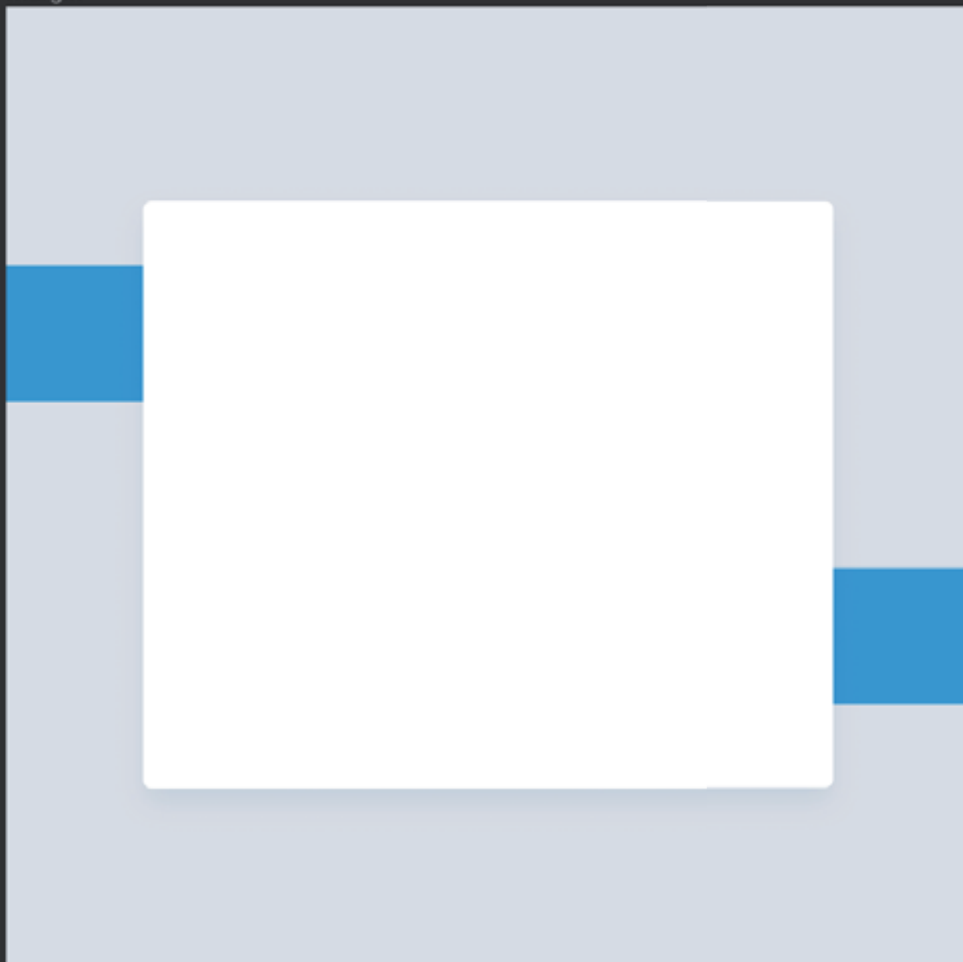


Ещё один способ мерить расстояния

Прямоугольники сами по себе — идеальный измеритель. Их удобно плющить до нужной формы, а затем мерить ими расстояния. Такие прямоугольники я называю ровнялками.

Например, самый быстрый способ задать отступ в 64 пикселя — сделать вспомогательные прямоугольники и по ним подстроить ширину основной фигуры. Затем, удалить их или вынести за пределы фрейма. Ещё более изощрённый способ ровнять объекты — обернуть ровнялки в компоненты и вставлять нужные величины из панели компонентов. Например, я знаю, что от основного блока до нижнего края макета у меня 120 пикселей. Я ввожу 120 в поиске по компонентам и вставляю квадрат такого размера, которым меряю расстояние до края макета.

margin



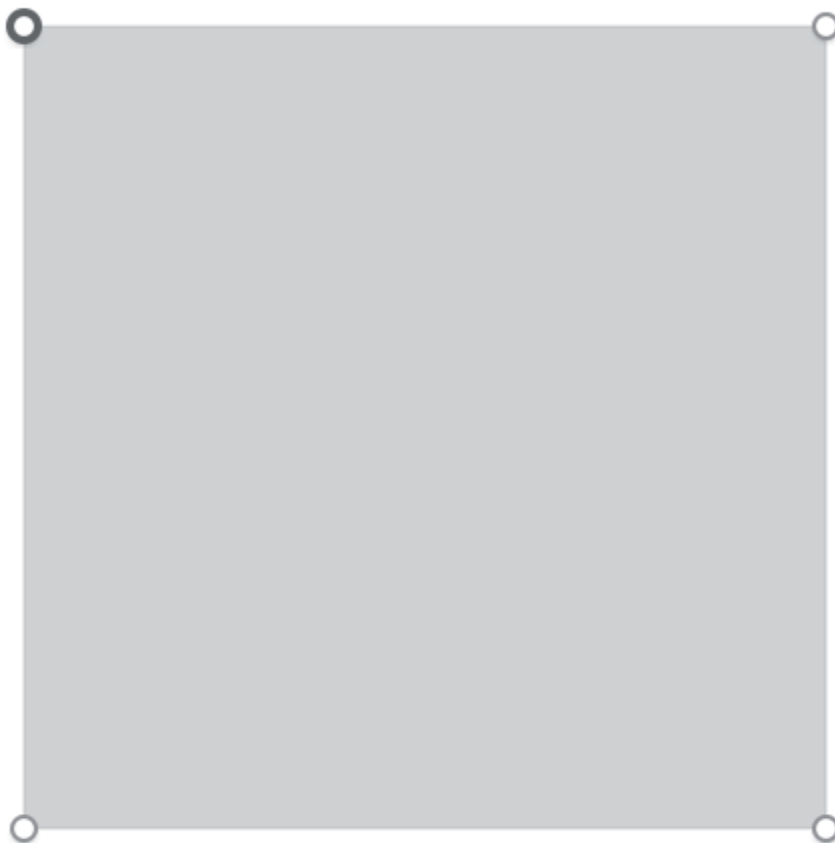
Режим редактирования

Как и в Скетче, если выделить фигуру и нажать **Enter**, мы перейдём в режим редактирования. Используем **Esc** или **Enter** чтобы выйти из него. При ховере на фигуру, Фигма наглядно показывает штриховкой, что мы находимся в этом режиме.



Когда переходим в аналогичный режим в Скетче, одна из точек, формирующих фигуру, выделена. **Tab** позволяет переключаться между ними и двигать их с клавиатуры.

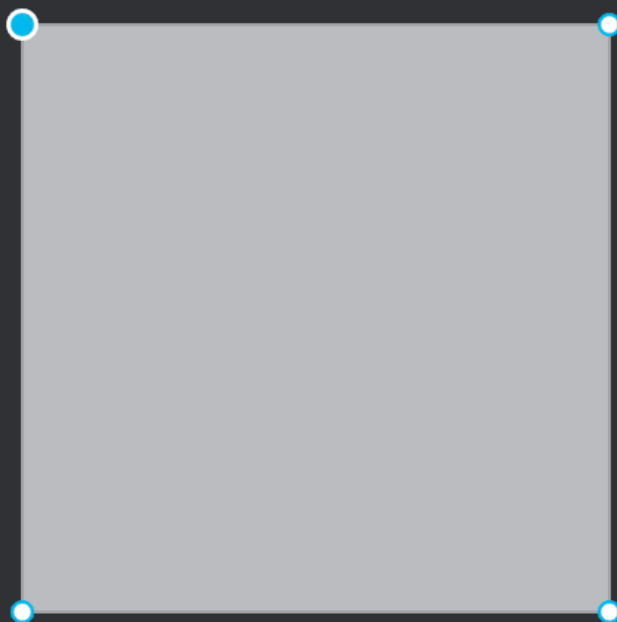
Выделенная точка в Скетче имеет более жирную обводку:



В Фигме точки надо выделять вручную, к сожалению, через **Tab** как в Скетче они не переключаются.

Если точка выделена, её можно двигать через **стрелки** и **Shift + стрелки**.

Выделенная точка залита цветом:



Прямоугольник можно превратить в другую фигуру, например, в трапецию. Для этого передвигаем его точки в режиме редактирования. В этом случае у него безвозвратно исчезнут ручки скругления. Это происходит, потому что Фигма перестанет воспринимать его как объект типа **Rectangle**, а будет считать его векторной фигурой произвольной формы.

В режиме редактирования можно добавить прямоугольнику новых точек и вылепить из него новую фигуру.

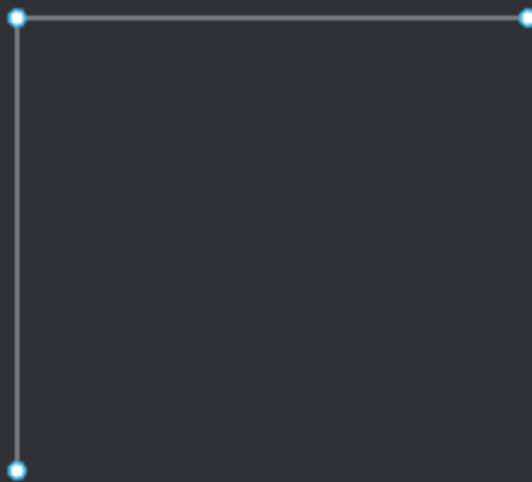
Также прямоугольник можно разорвать в путь. Для этого нужно удалить из него любую точку.

В Скетче для разрыва пути используется инструмент **Scissors**.

Скетч: удаление точки не разрывает фигуру, а уменьшает её площадь. Квадрат, которому удалили точку в Скетче, станет треугольником.

Фигма: Удаление точки разрывает фигуру. Вместе с ней удалятся прилегающие к ней отрезки.

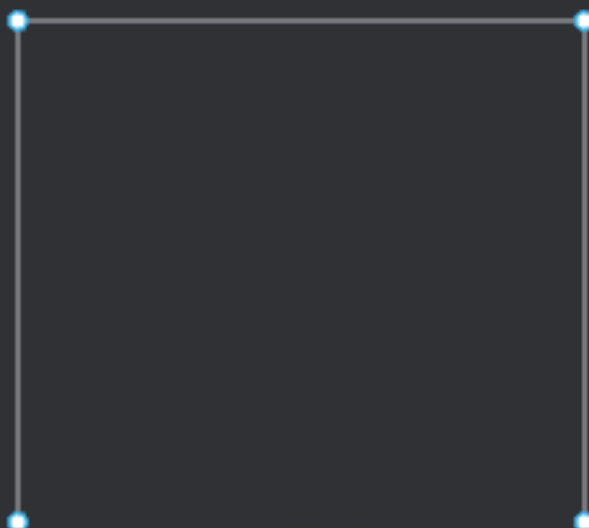
Выделяем точку, удаляются точка и две стороны фигуры:



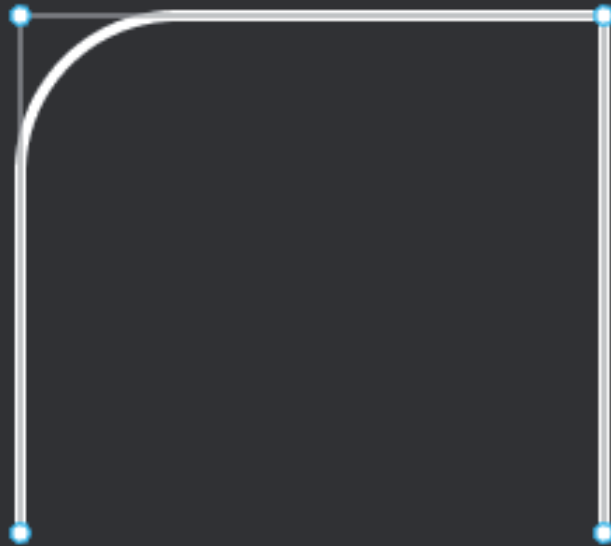
В Фигме не нужен инструмент **Scissors**. Если нужно удалить только одну сторону, создаём точку по её центру. Фигма при ховере подсказывает, где центр отрезка:



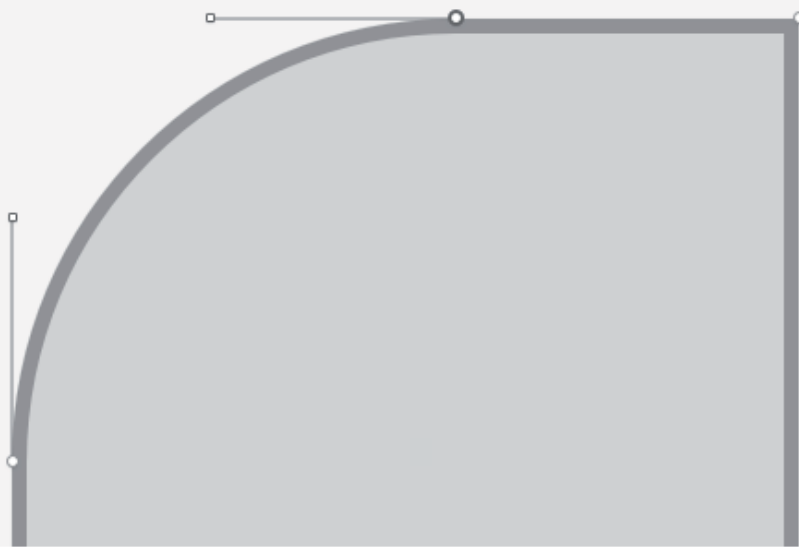
Удаляем точку, **Delete**. Остался векторный скелет из точек и отрезков. При этом у левой верхней точки осталось скругление, которого не видно из-за того, что у векторной фигуры нет обводки.



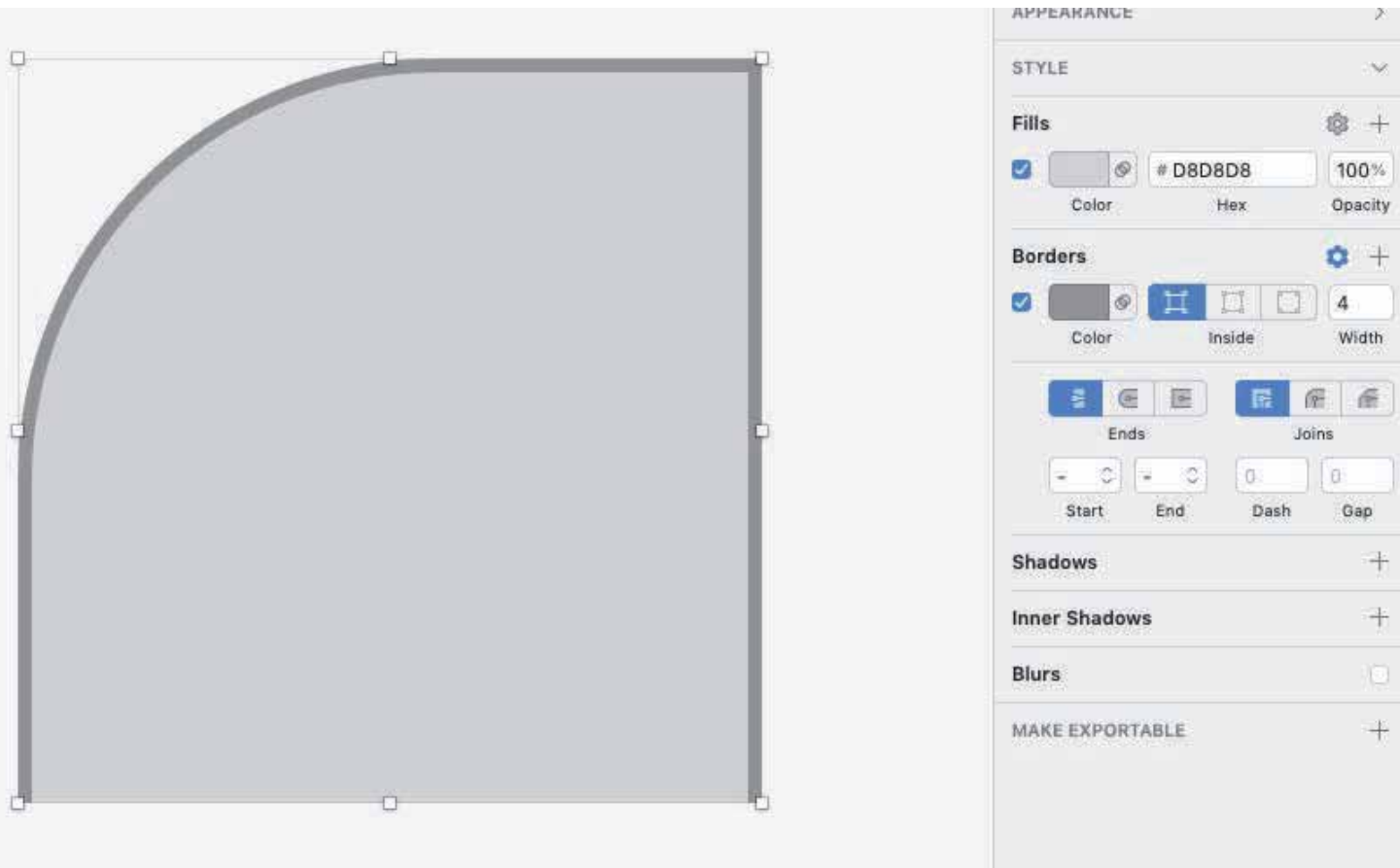
Таким образом, мы преобразовали фигуру типа **Rectangle** в разомкнутый путь. Задаём **stroke**, чтобы увидеть скругление:



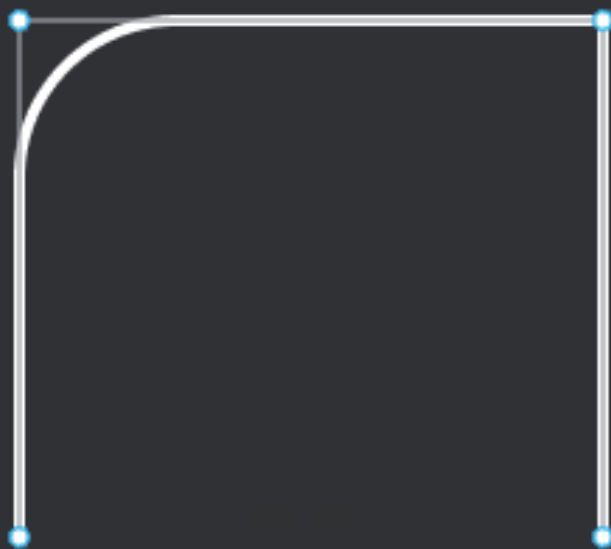
Скруглённый угол остался. Это значит, что каждая точка векторного пути хранит в себе информацию о степени скругления, даже если шейп разорван. В Скетче после разрыва шейпа скругление превращается в отдельную кривую, лишая нас возможности редактировать её через значение **Corner Radius**. Работа со скруглениями в Фигме реализована лучше.



В Скетче даже у разомкнутых путей может быть заливка. Применяем **Fill** в Скетче:



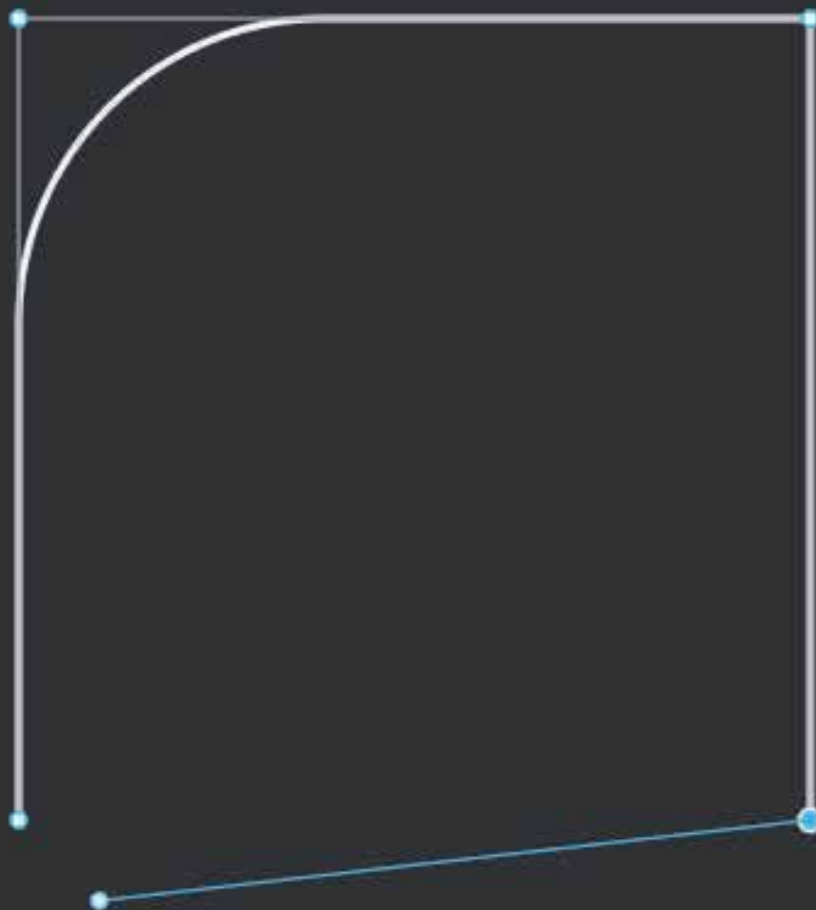
Аналогичная ситуация в Фигме: заливка есть в настройках, но не видна, поскольку фигура разомкнута.



Теперь Фигма воспринимает такой объект как векторный путь, или векторную сеть. У пути нет понятия внутреннего и внешнего, поэтому обводка может быть только в режиме **Center**, вне зависимости от того, какой режим выставлен в настройках.

Аналогично Скетчу: Если взять **Pen Tool**, **P**, навести на одну из конечных точек, в курсоре появится круг. Это значит, что при нажатии мы будем продолжать рисовать линию с этой точки.

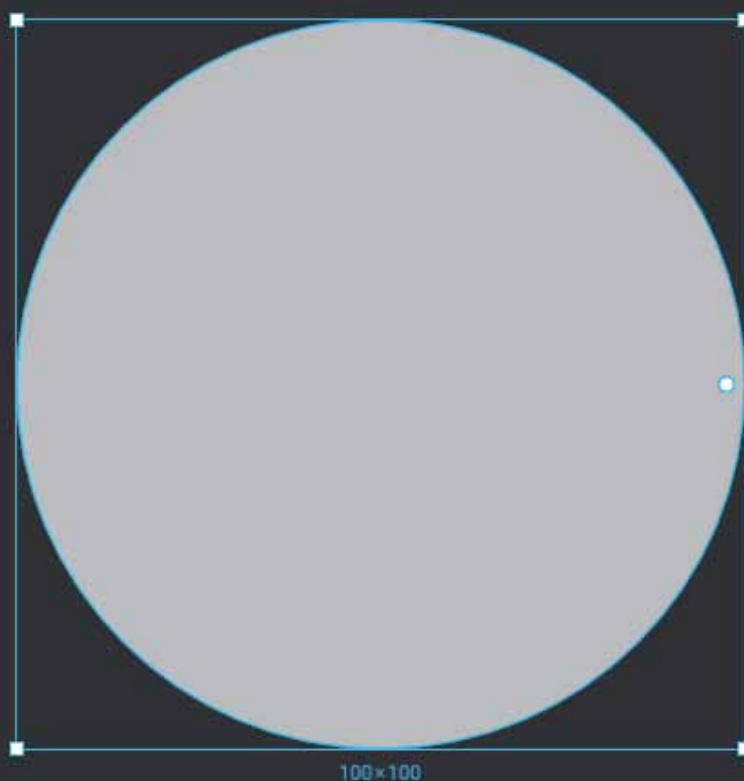
Кликаем в точку, кликаем в другую напротив и таким образом замыкаем фигуру обратно:



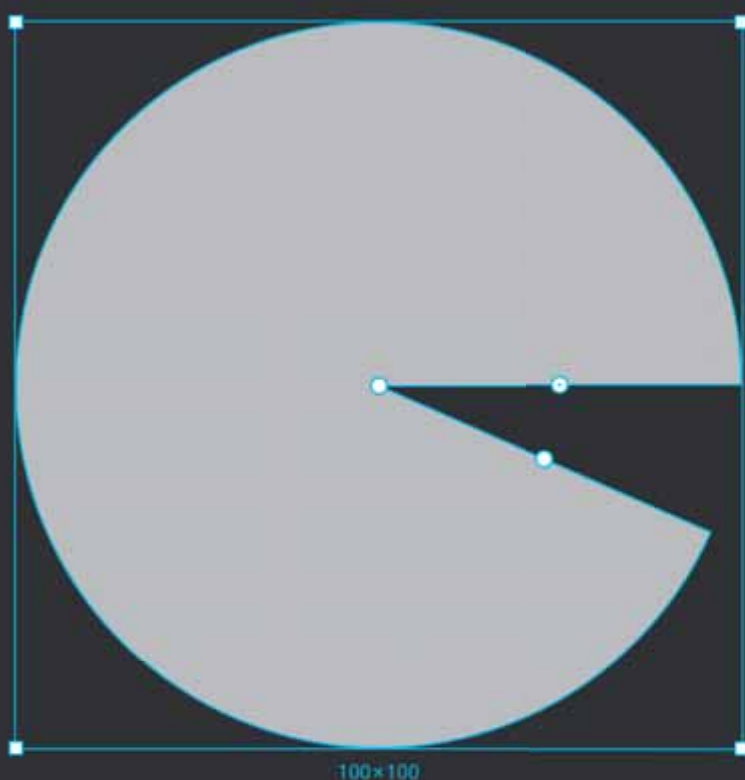
Окружность, овал и пайчарт

Клавиша: **O** Ellipse, **O**val

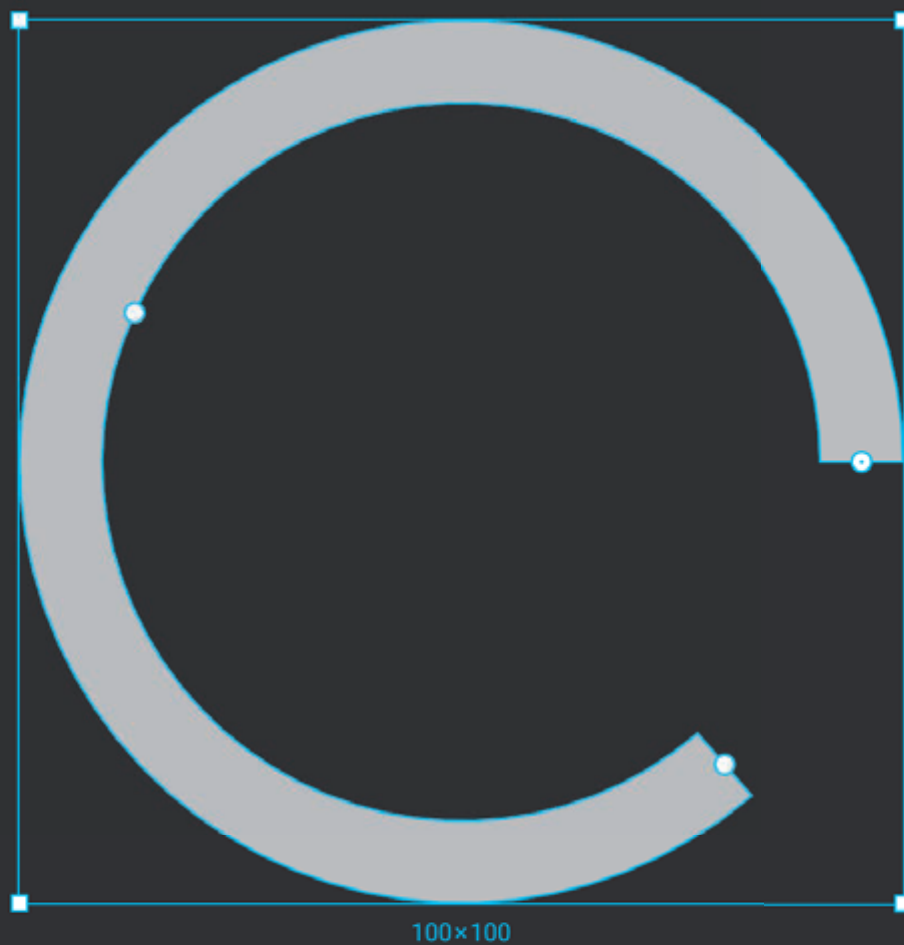
Рисуем круг с зажатым **Shift**, либо кликом в любое место:



В отличие от Скетча, в Фигме окружности и овалы можно легко размыкать, не используя **Pen Tool** или ножницы. Для этого наводим на фигуру и тянем круглую ручку в правой части вверх или вниз. Так мы меняем свойство **Sweep**.

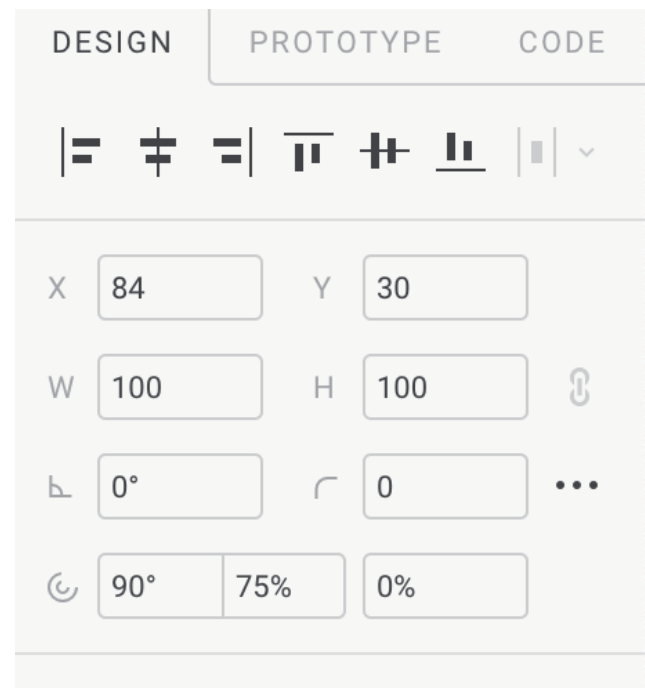


Когда мы задали значение, получившийся пайчарт имеет два радиуса с ручками, которые можно подстраивать в нужные значения, а также центральная точка, которая позволяет вырезать сердцевину. Перетаскиваем центр вниз, вырезая из круга середину:

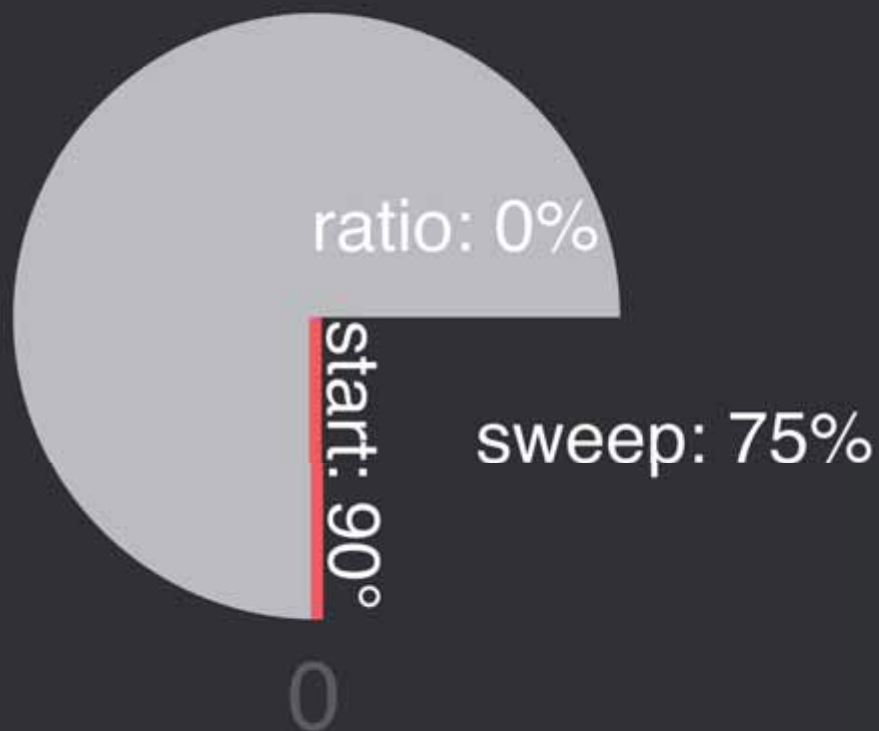


За работу с пайчартами отвечают три свойства, которые есть в панели выделенной окружности:

- **Start** — угол начала в градусах, изначально 0. Исходный градус — на 3 часа.
- **Sweep** — процент заполненности от 0 до 100%.
- **Ratio** [рэйшио] — размер пустоты по центру в процентах от радиуса. При 0 пайчарт будет заполнен по центру, при 100 заливки видно не будет, останется только stroke.



Ещё один пример: **start** с трёх часов повернут на 90°, **sweep** занял 3/4 окружности. Он высчитывается из положения **start**. **Ratio** не используется.

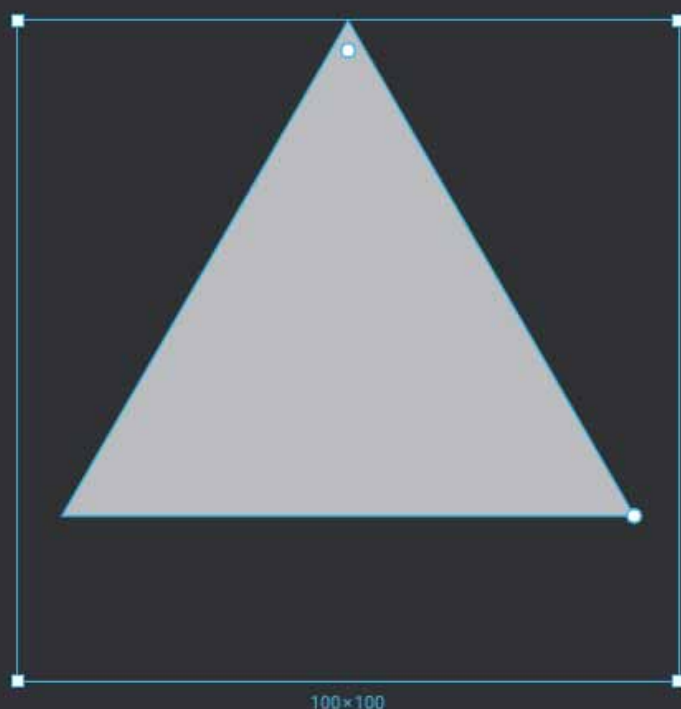


В Скетче пайчарты можно рисовать либо через плагины, либо через обводку с пунктиром. В Фигме значительно легче задавать нужные значения пайчартов в визуальном режиме.

Треугольник

Инструмент **Polygon** позволяет рисовать многоугольники.

Не имеет ровной клавиши, можно вызывать **Cmd + /**, **pol**. Чтобы треугольник был равносторонним, при его растягивании зажимаем **Shift**.



Получившийся равносторонний треугольник имеет две круглые ручки: внутри сверху и справа на углу.

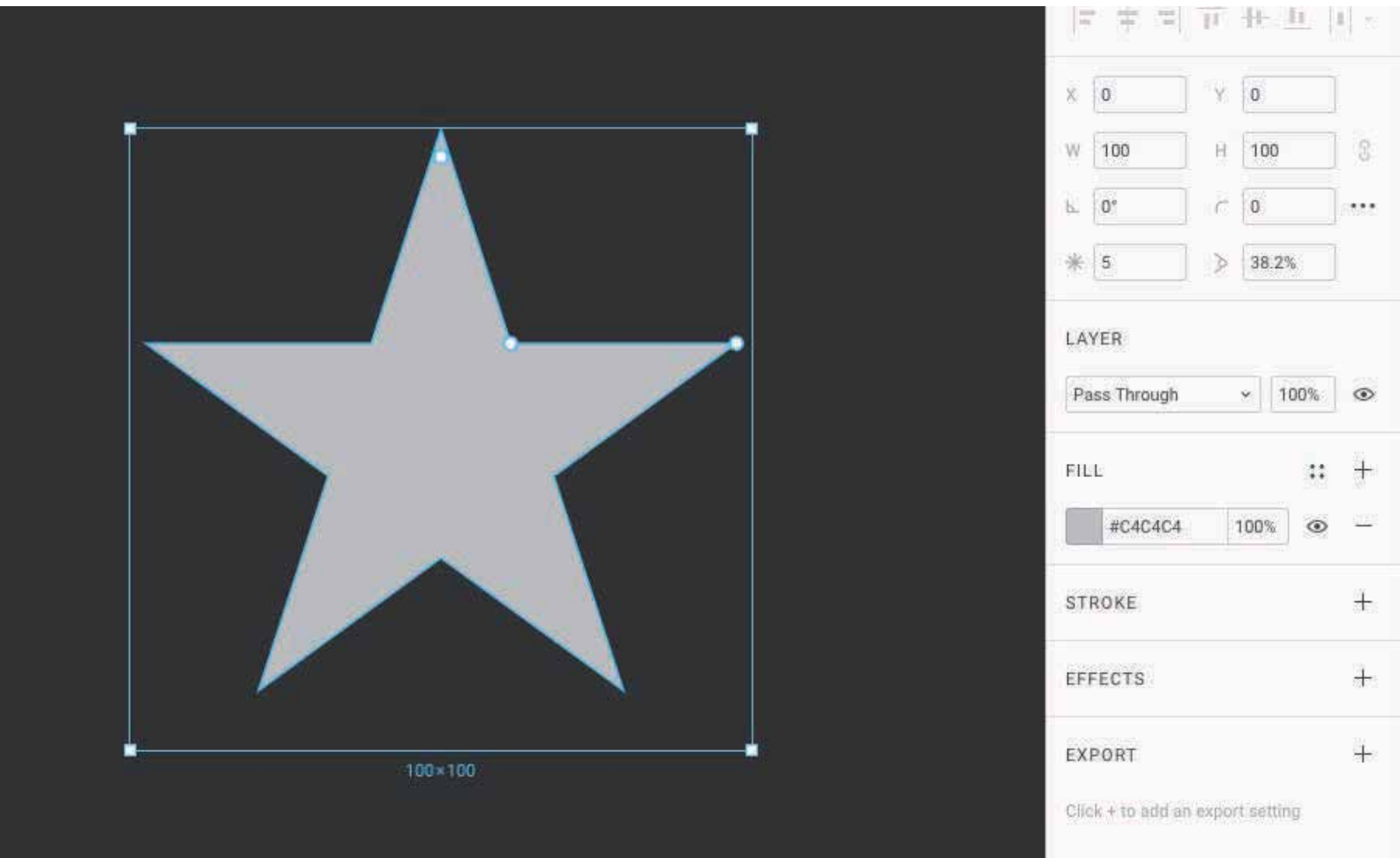
Верхняя ручка позволяет **скруглять углы** треугольника и работает так же как на прямоугольниках, однако если нужно скруглить только один угол, надо заходить в режим редактирования.

Угловая ручка позволяет задавать нужное **количество углов** в интервале от 3 до 60. При 60 углах многоугольник мало отличается от окружности. Также задавать углы можно в **Count**.

Звезда

Инструмент **Star** позволяет рисовать лучевые многоугольники.

Не имеет ровной клавиши, можно вызывать **Cmd + /**, **sta**.

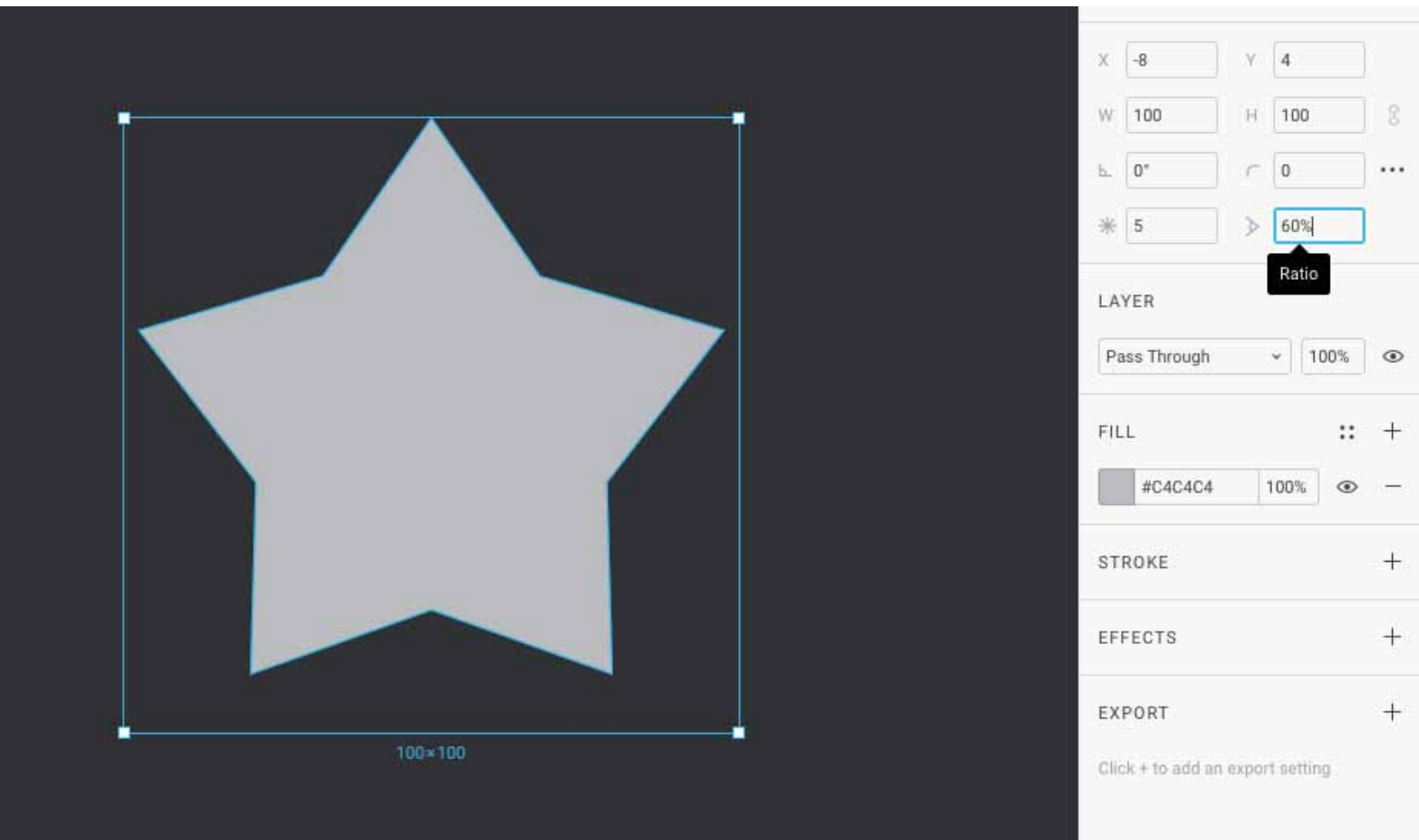


Помимо стандартных, у звезды есть 3 параметра:

- **Corner Radius** — степень скругления углов
- **Count** — количество лучей
- **Ratio** — степень остроты лучей

Ручка в верхнем луче позволяет скруглять углы всех лучей.

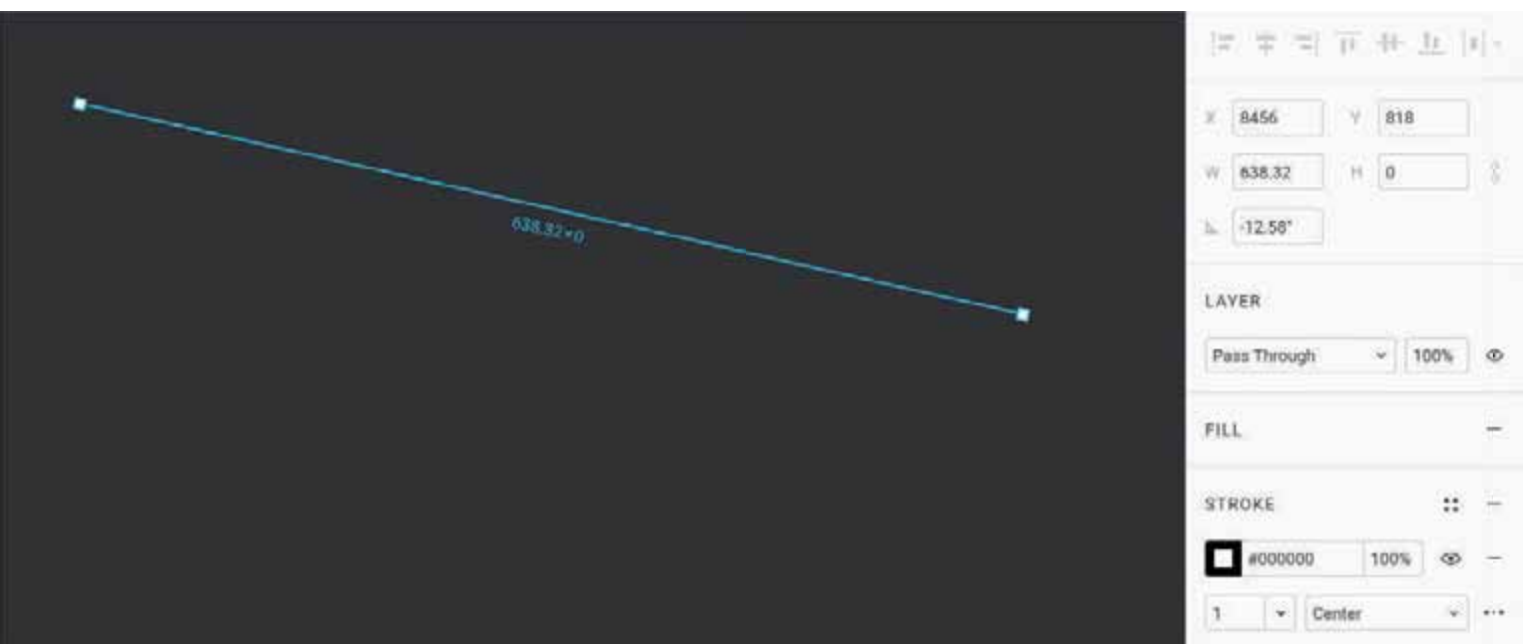
Ручка в углу между верхним и правым лучами позволяет задавать **ratio**:
будет ли звезда остроконечной или пузатой.



Линия

Line Tool, L

Инструмент позволяет при помощи перетаскивания создать векторный отрезок. Чтобы нарисовать линию, выбираем место начала,жимаем левую, тащим. Там где надо, отпускаем.



В Скетче линия имеет **Start** — координаты начала и **End** — координаты конца, а также длину и угол наклона, который не зависит от реального угла наклона начерченной линии. Если провести линию на определённый угол, в Скетче в поле угла останется 0, а в Фигме оно будет соответствовать этому углу.

Две пары координат усложняют работу с такой линией, если нужна точность.



Start	1109,5 X	116,5 Y
End	1545,51 X	211,67 Y
446,28	0 °	

APPEARANCE ▾

No Layer Style ▾

Opacity (Normal) ▾ 100%

STYLE ▾

Fills

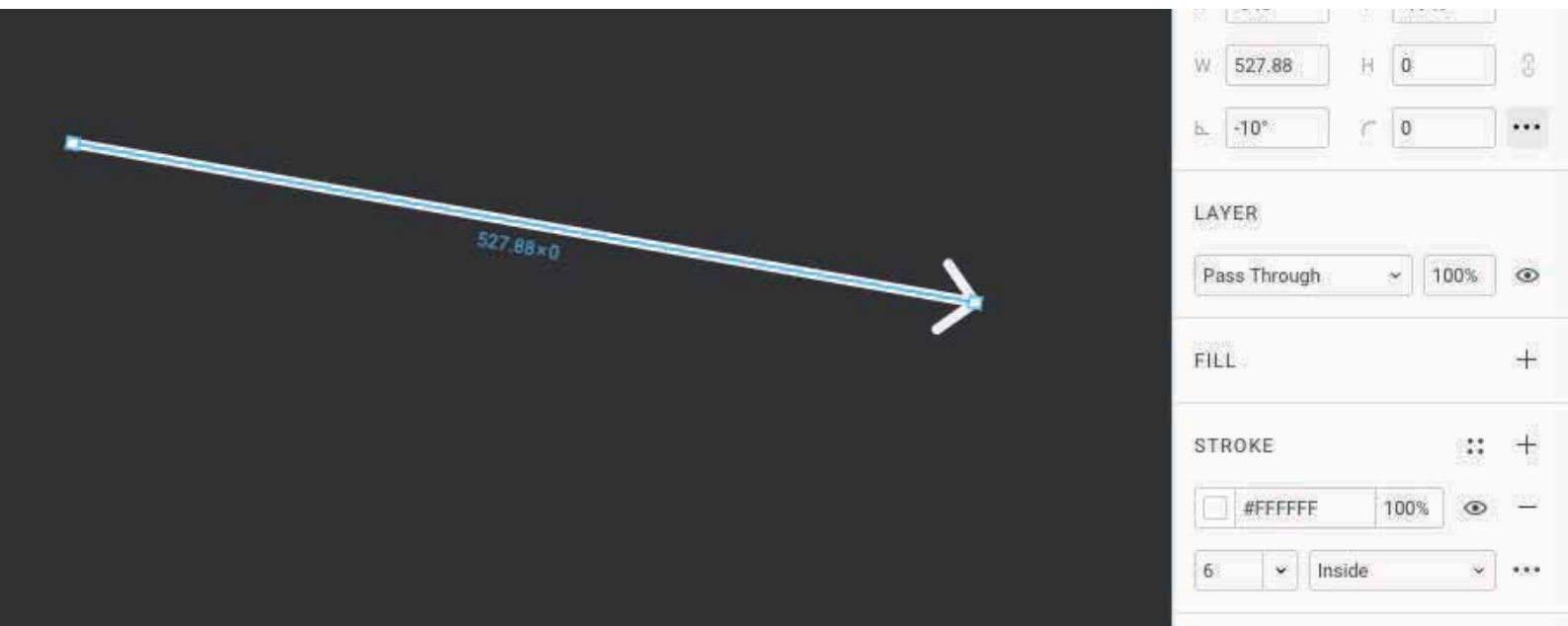
Разработчики Фигмы не наступили на грабли Скетча и сделали только одну пару координат для начала линии. В поле **Width** линии можно задать длину, а в **Angle** угол наклона.

Некоторые используют **Line Tool** для того, чтобы делать отбивки между строками в таблицах. Однако, их проще контролировать через **Rectangle**.

Стрелка

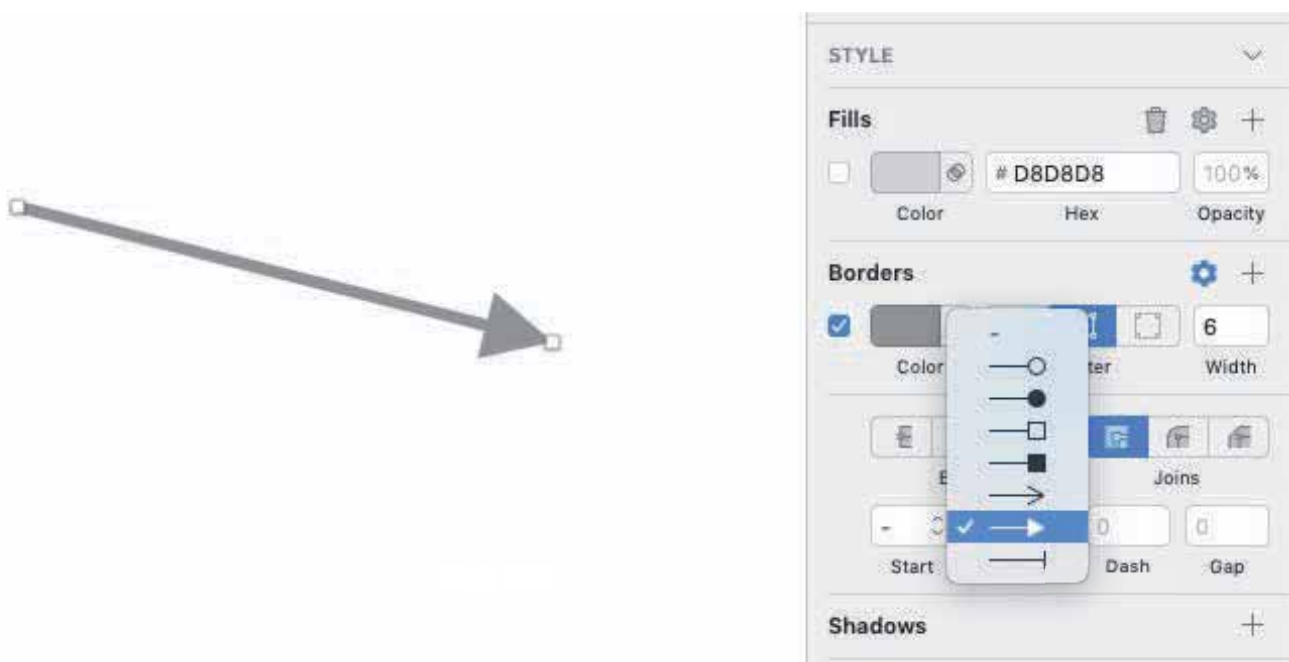
Arrow Tool, **Shift + L**

Инструмент работает аналогично **Line Tool**, единственная разница — на конечной точке имеется стрелка.

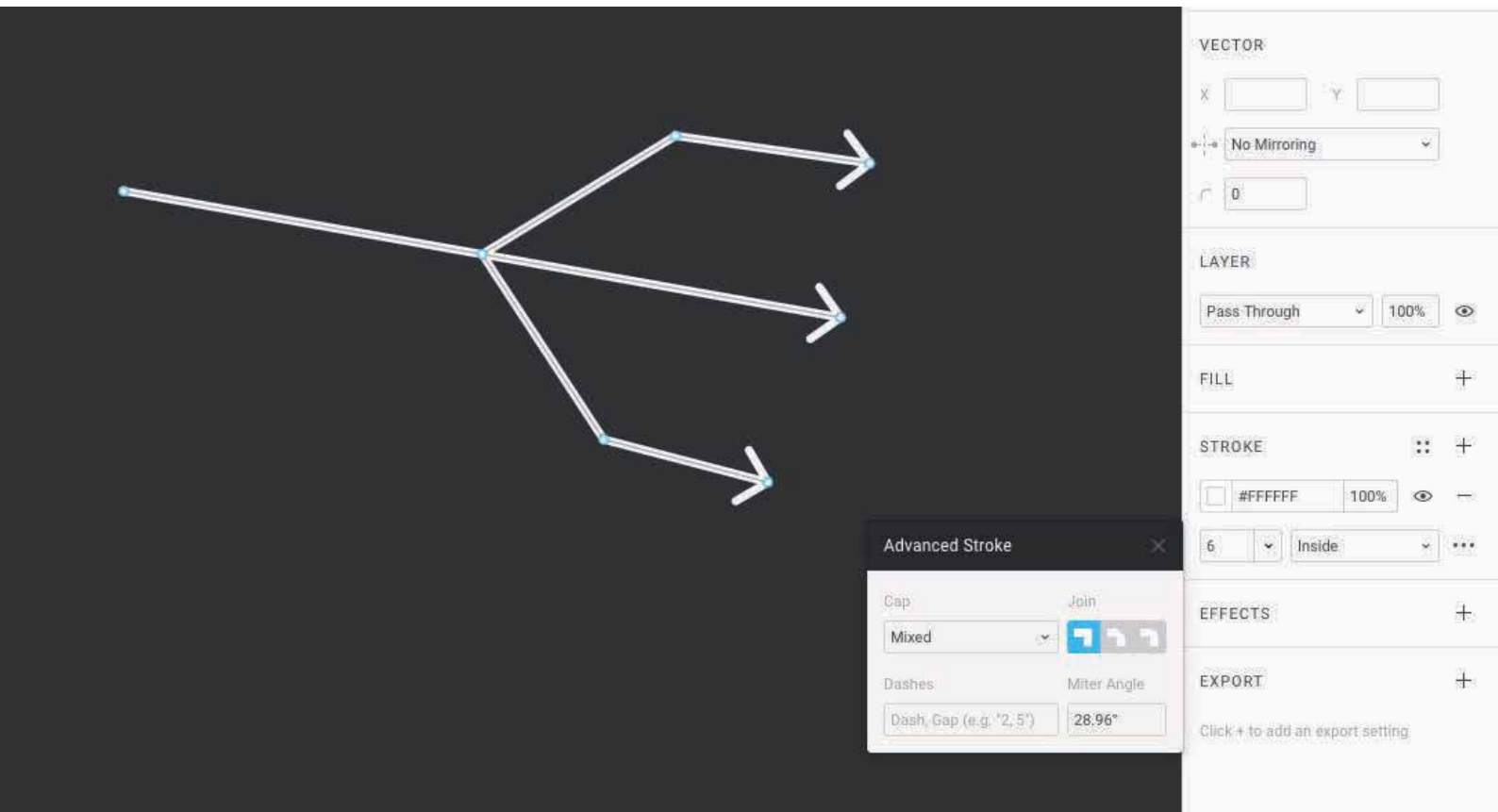


Как и в Скетче, стрелка на конце — это определённое свойство. Однако, в Скетче стрелка — это свойство всей линии, а в Фигме — это свойство отдельной точки.

Как настраивается стрелка в Скетче: выбираем тип стрелки в поле **End**.



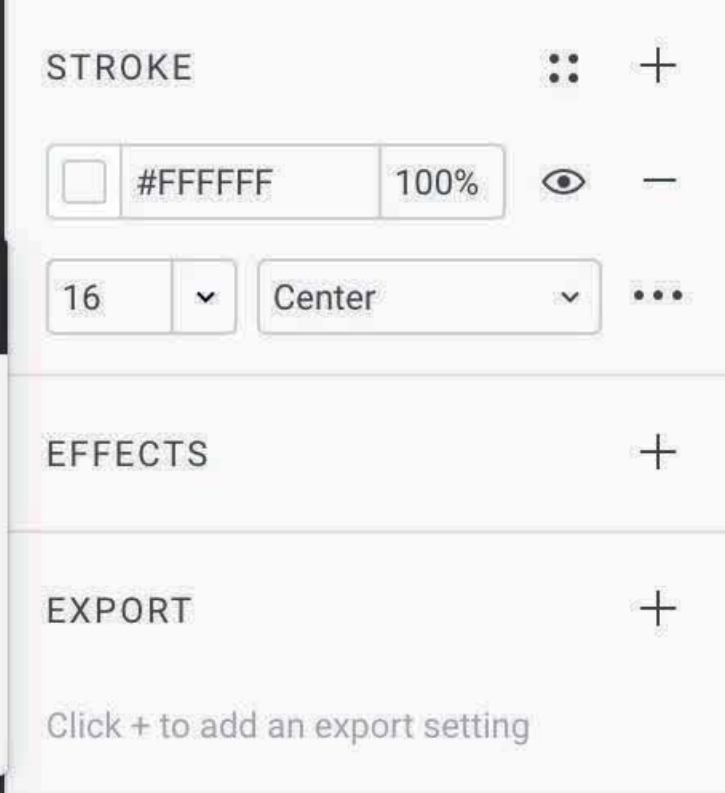
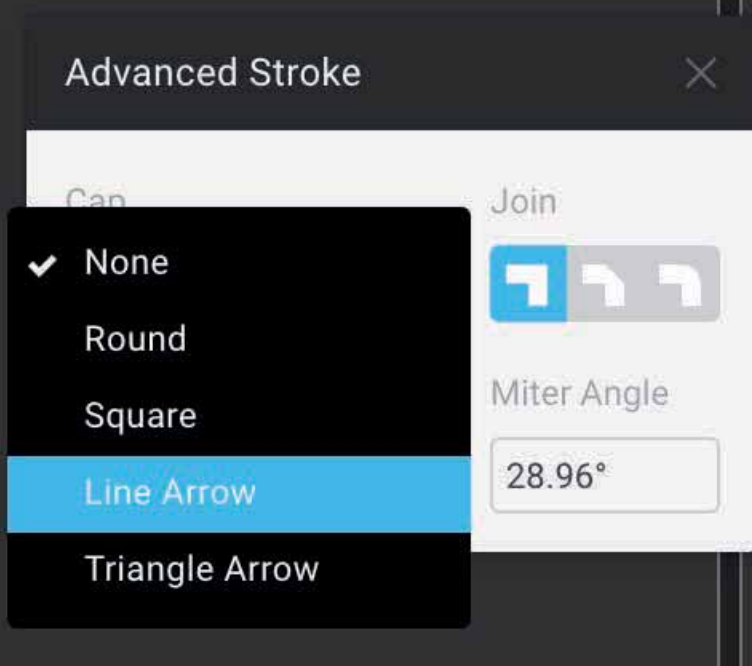
В Скетче линия может иметь начальную и конечную стрелку, а в Фигме стрелку может иметь любая конечная точка векторной сети. Следовательно, может быть стрелка, у которой одно начало и несколько наконечников:



Наконечник точки можно отключать, превращая стрелку в обычную линию. Либо наоборот, можно нарисовать кривую пером **Vector Tool**, **V** и настроить ей стрелку на любом из концов.

В Скетче можно копировать стиль стрелок, **Opt + Cmd + C** и передавать их другим линиям, **Opt + Cmd + V**.

В Фигме этого нельзя делать, поскольку таким образом мы передадим только информацию о цвете и ширине линии. Наконечники придётся настраивать индивидуально.



Чтобы настроить стрелку, нужно в режиме редактирования **Enter** выбрать нужную точку и развернуть меню **Advanced Stroke**. Это меню выглядит как три точки в блоке **Stroke**.

В нём есть меню **Cap**, где можно выбрать тип наконечника:



- **None**
- **Round** — закруглённый
- **Square** — квадратный, похож на **None**
- **Line Arrow** — контурная стрелка
- **Triangle Arrow** — треугольная стрелка



9. Перо

[Проект в Фигме](#) →

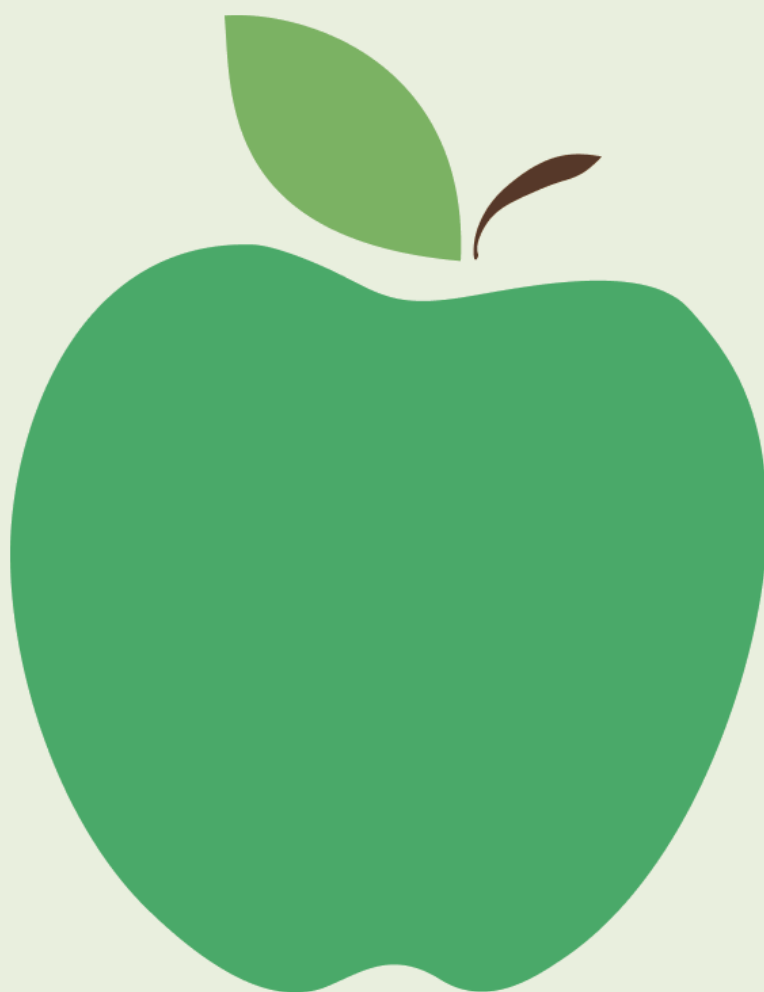
Pen Tool, **P** — важнейший инструмент для рисования кривых линий и сложных непредсказуемых форм при помощи кривых Безье.

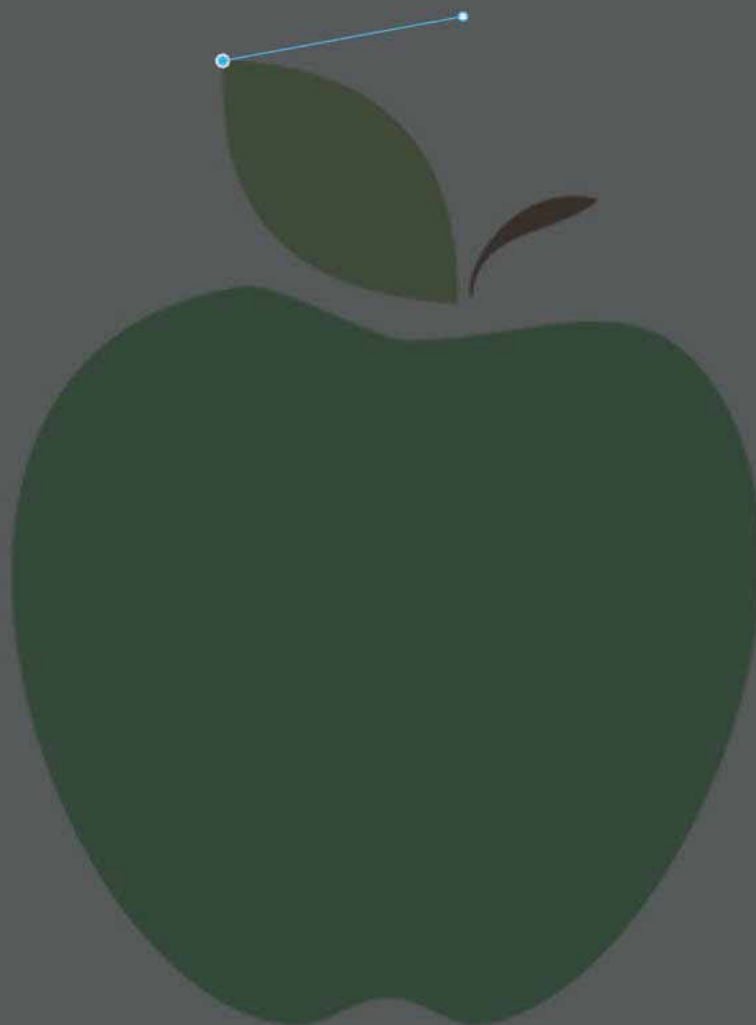
В Скетче называется **Vector Tool**, **V**.

Работа пера в Фигме больше похожа на работу в Иллюстраторе, чем в Скетче, и это хорошая новость.

Чтобы научиться использовать перо, обрисуем яблоко.

Источник вектора: freevector.com/apple-vector-icon

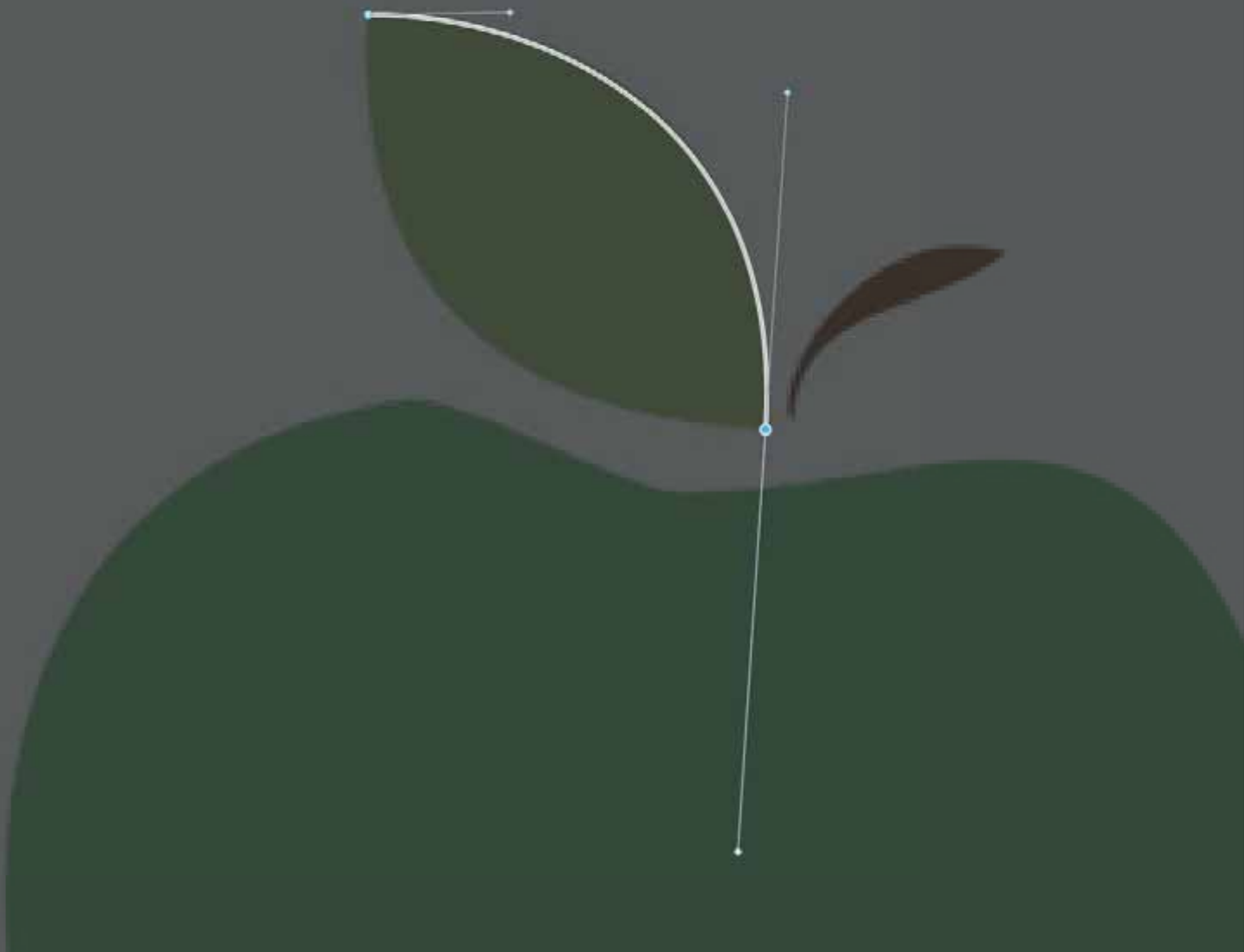




Чтобы изображение не мешало видеть линию контура, сделаем его на 20%, **2**. Также выберем белый цвет линий пера.

1. Выбираем Pen Tool, **P**.
2. Начнём с листа. На верхнем углу листа левым кликом создаём точку, из которой будет тянуться первая кривая. После того, как поставили точку, можно подкорректировать её положение **стрелками**. От точки до указателя тянется шлейф, который подсказывает, как будет выглядеть линия, если её зафиксировать в этом месте.

Наша задача — провести кривую от первой точки до того места, где её придётся прервать.

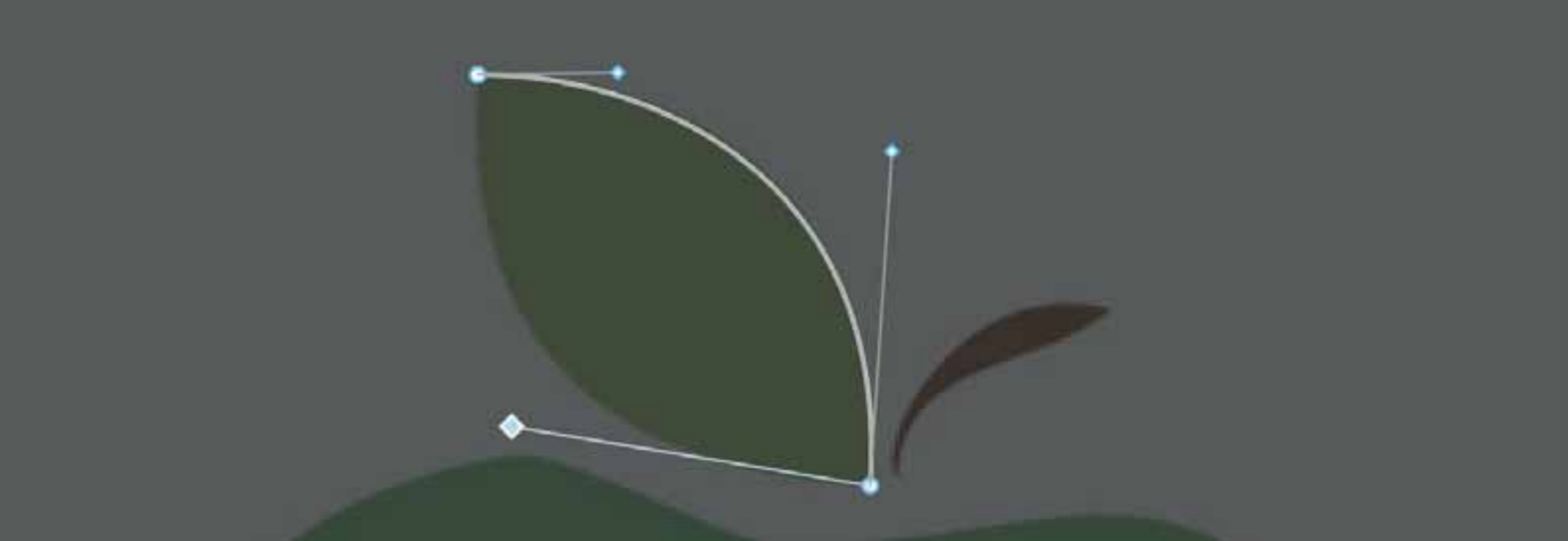


3. Кликаем на нижний угол листа и тянем вниз. Из второй точки появляются тонкие белые рычаги — усы.

Усы символизируют скорость, из которой линия выходит из точки к центру линии. Чем усы длиннее, тем сильнее они влияют на искривление линии. Первый ус покороче и выходит горизонтально. Не отрывая указателя, тянем за второй ус вниз и немного левее. Таким образом, верхняя его половина смещается вправо, прогибая кривую. Отпускаем указатель. Линия фиксируется. От точки до указателя снова тянется шлейф.

В этот раз он может искривляться, потому что у начальной точки новой кривой уже есть *скорость*. За один клик мы установили сразу два рычага: для окончания первой кривой и для начала второй.

4. Теперь нужно загнуть ус так чтобы он направил следующую кривую туда же, куда идёт контур листа. Для этого мы берём нижний ус и тащим его влево. Подбираем нужный угол и делаем ус той же длины, что и его вертикальный собрат.

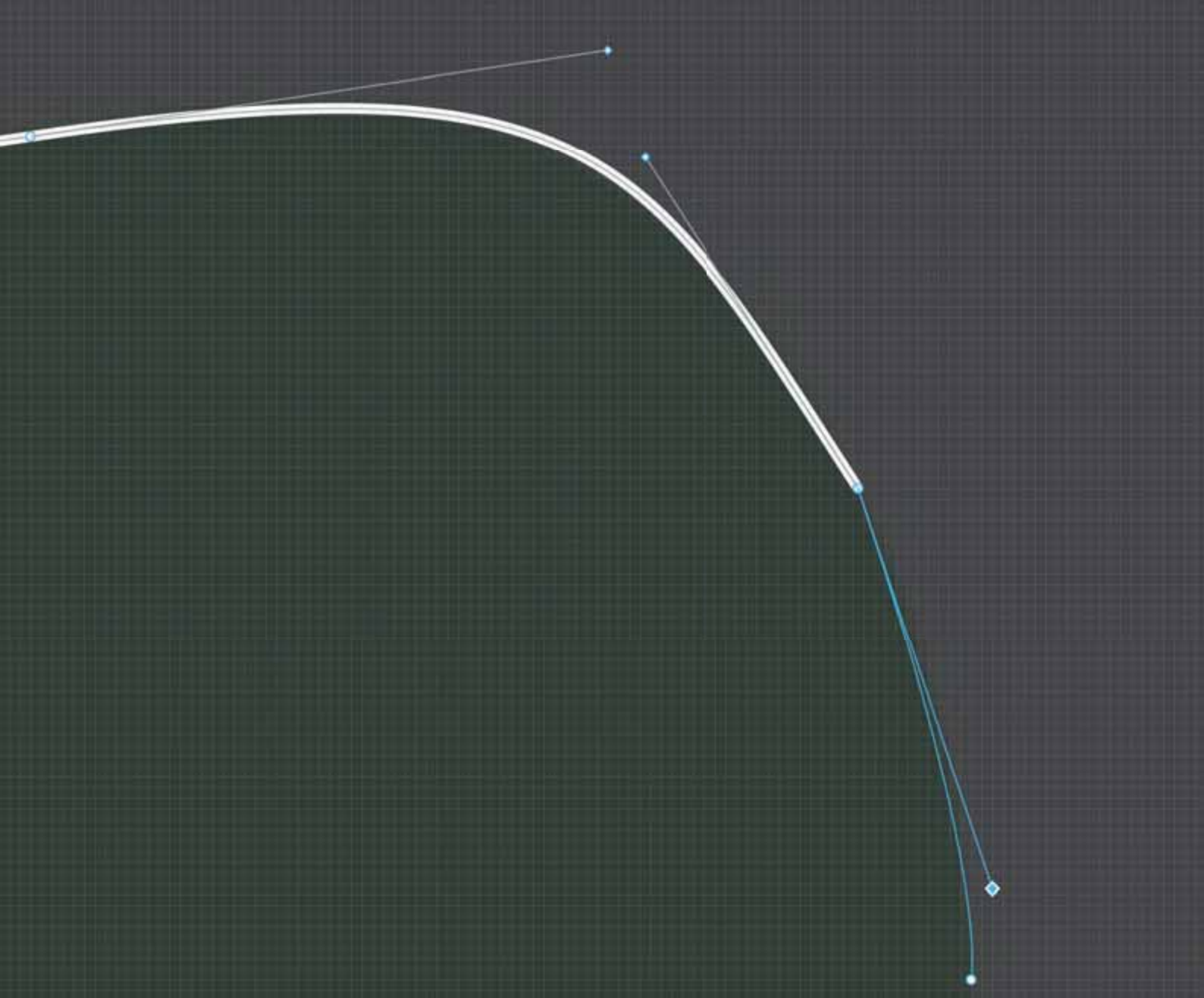


Ломаем усы при помощи **Opt**

Мы дошли до важного преимущества пера в Фигме и Иллюстраторе перед **Vector Tool** в Скетче. На шаге 3 мы зафиксировали ус, рычаги которого стоят параллельно, а на шаге 4 скорректировали нижний рычаг. Это можно было бы сделать удобнее: кликнуть точку, растянуть из неё усы, подстроить верхний ус, держась за нижний. Затем, не отрывая указателя, зажать **Opt**. Пока клавиша зажата, верхний перестанет двигаться, а нижний можно тянуть дальше. Таким образом, при помощи клавиши мы за один клик можем задать поведение двух кривых: как будет вести себя конец первой и начало второй.

В Скетче **Opt** так не работает, а значит, на каждом шаге нужно корректировать уже зафиксированный второй ус вручную.

5. Замыкаем контур листа, кликая в первую точку и вытягивая из неё вверх новую пару усов. Подстраиваем изгиб второй кривой за верхний ус.
6. Если требуется подкорректировать любой из усов, это можно сделать, если выделить любую точку в фигуре.
7. Выходим из режима редактирования, зафиксировав фигуру, **Enter**.



Частая ошибка: изломы линий

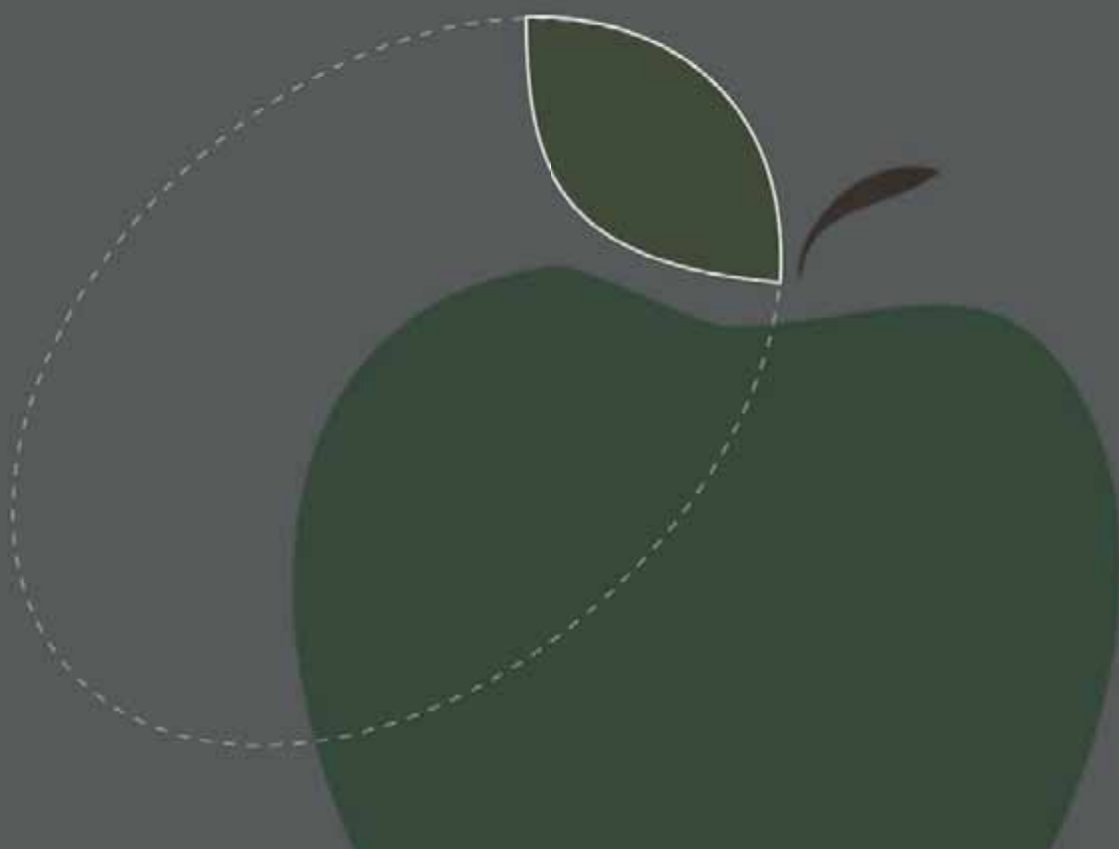
При обрисовке следует избегать тупых углов там где они не нужны. В скриншоте выше усы образуют тупой угол, который будет привлекать внимание в финальном контуре. Делаем линии плавными. Для этого усы должны формировать развёрнутый угол.

Минимализм и точность

Стараемся использовать минимальное количество опорных точек: это повышает скорость работы. При этом кривые должны оставаться точными, чтобы не искажать форму объекта, который мы обрисовываем.

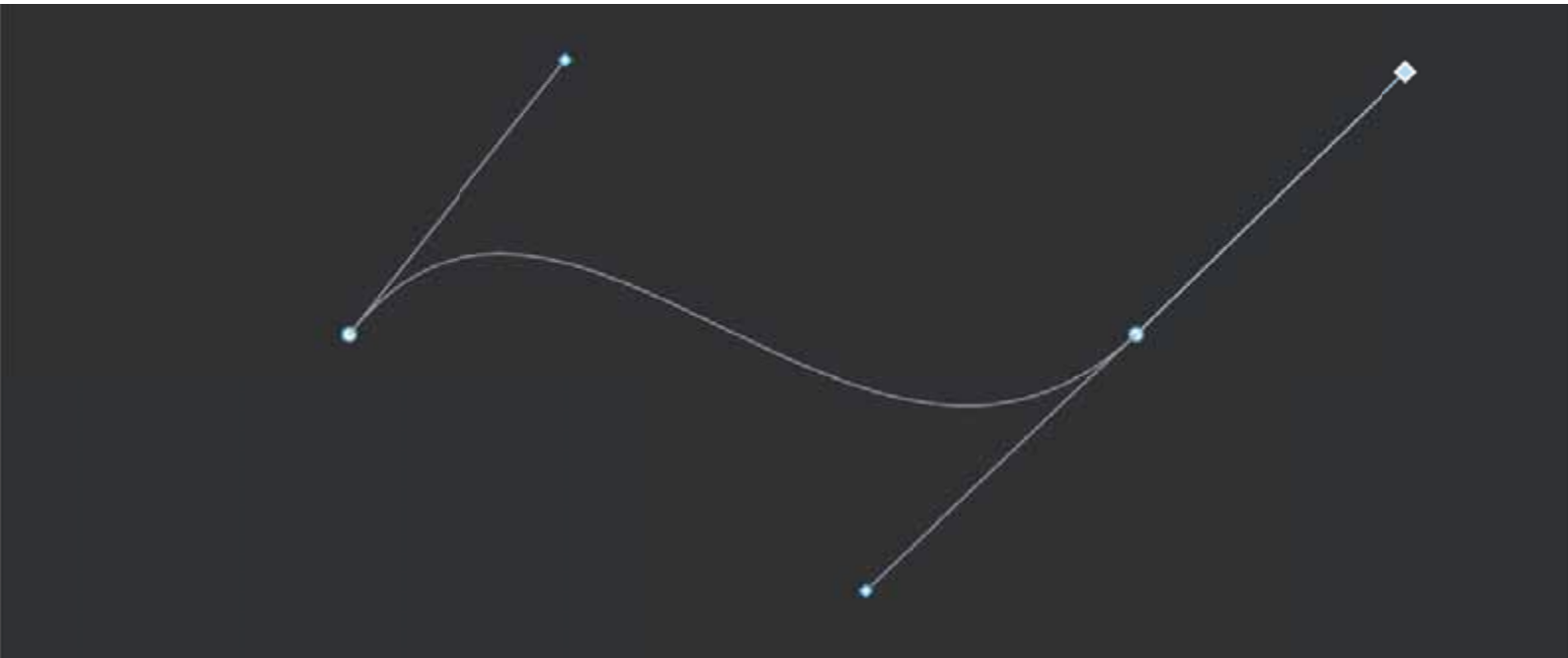
Обучаясь работе с пером, полезно представлять, как эллипсы могут вписаться в кривые.

Мы предугадываем, как может изогнуться линия, представляя её как часть эллипса, который мог бы быть вписан в нужный нам контур:

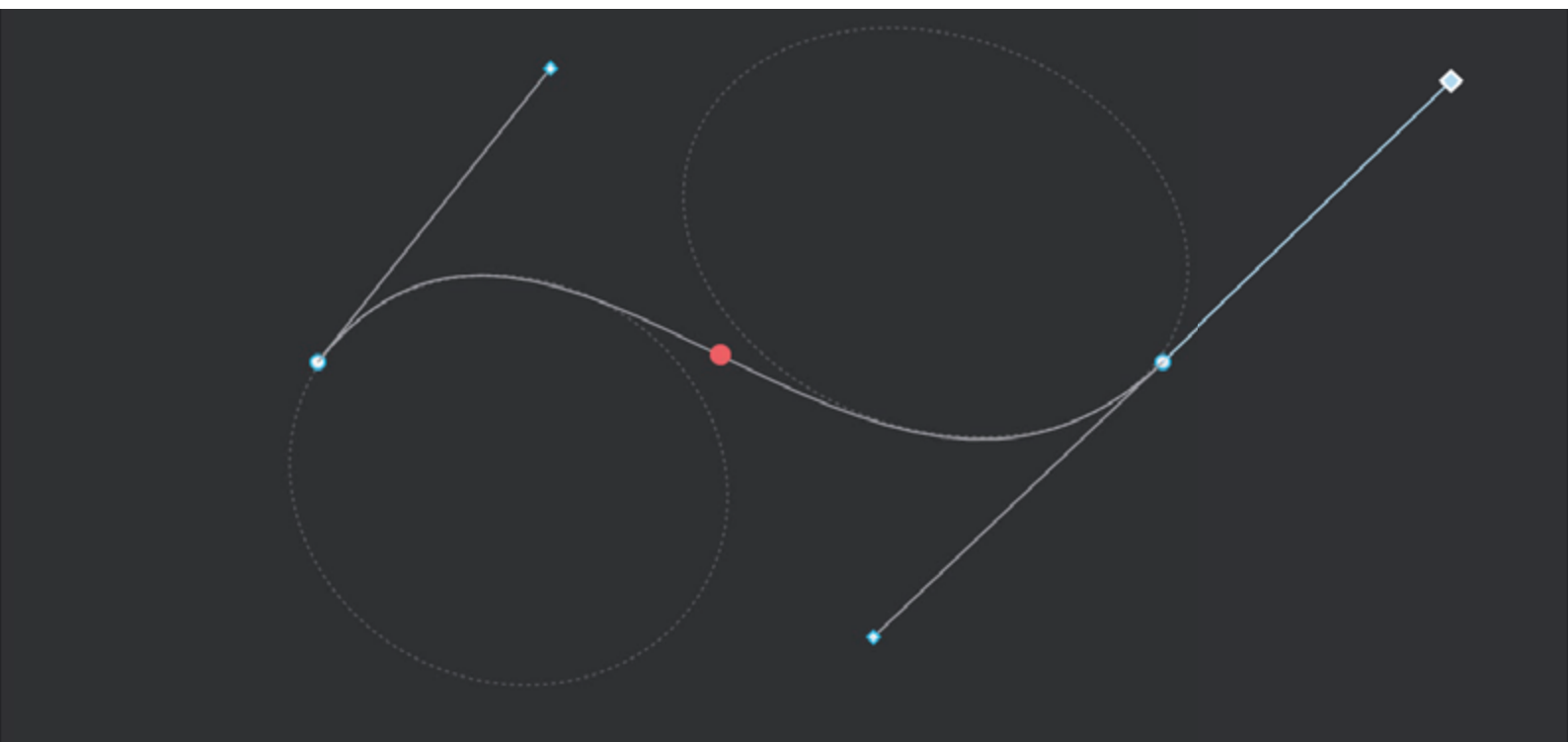


Это работает, потому что кривые Безье и эллипсы строятся по одним и тем же математическим правилам.

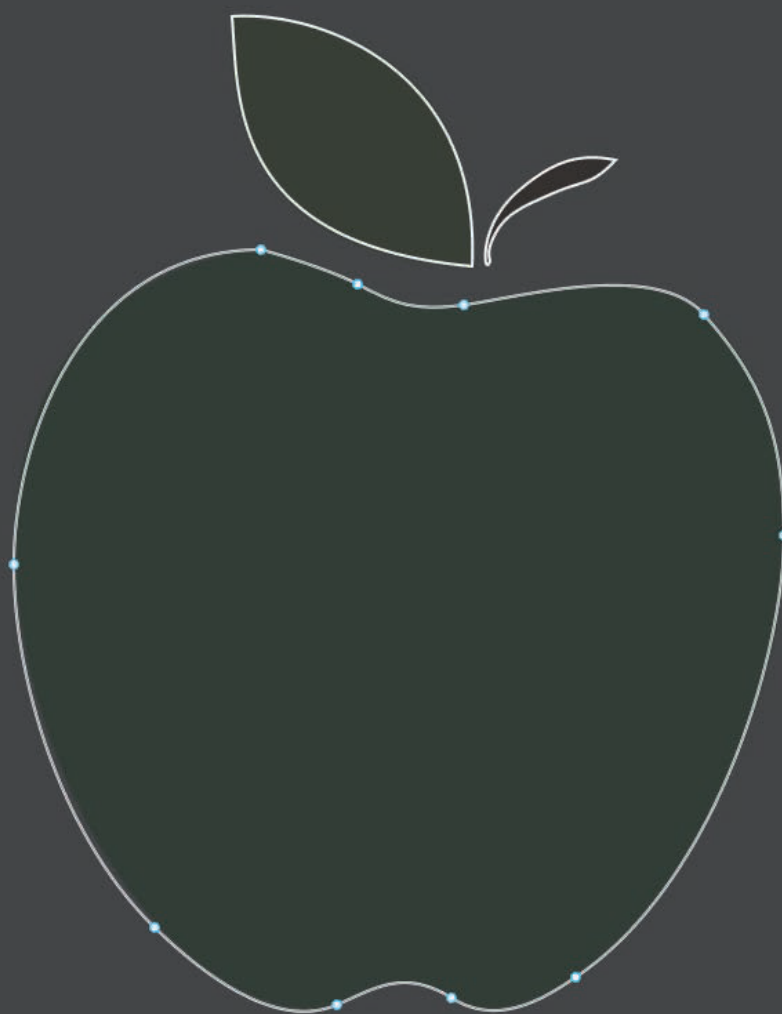
Если у начальной точки есть вытянутый ус, такую кривую не получится вписать в один эллипс, поскольку этот ус будет искажать её:



В этом случае мы можем наметить на кривой условное место, которое разделяет две простейших кривых. В каждую из них можно вписать эллипс.



Зная эти основы, мы можем обрисовать основную фигуру яблока:



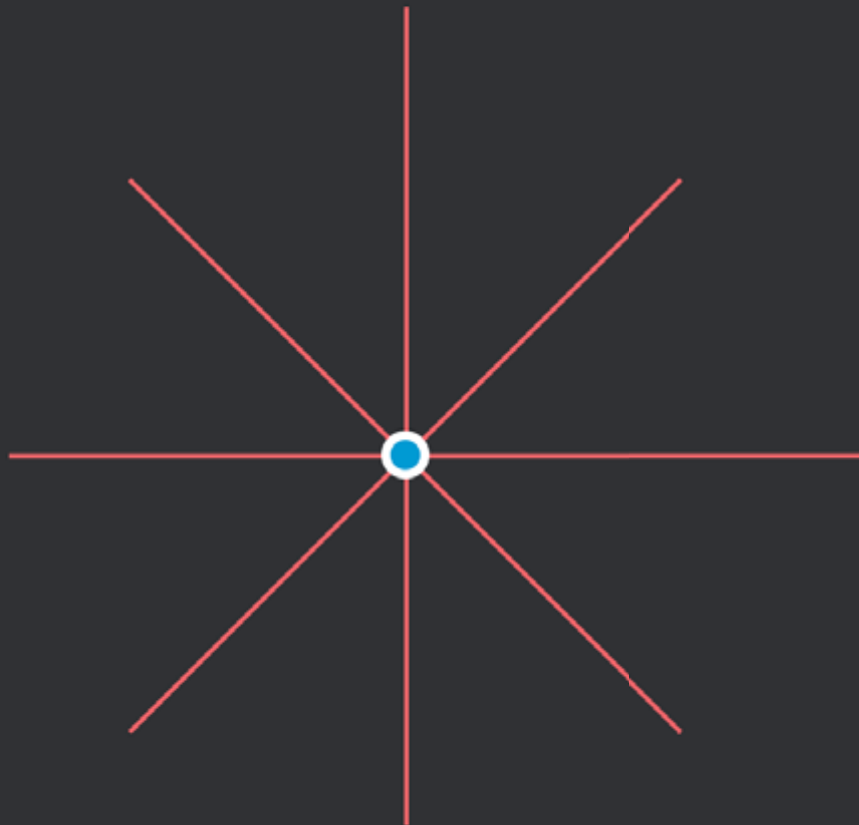
Shift и Snap to Geometry: чертим прямоугольник пером

Если нужно, чтобы линия была строго вертикальной или горизонтальной, можно зажать **Shift**. В таком режиме можно чертить геометрию. Нарисуем треугольник пером.

1. Выделяем перо, **P**. Ставим первую опорную точку. Смещаем указатель вправо.
2. Зажимаем **Shift**. Шлейф от точки до указателя станет красным и вытянется горизонтальной линией вправо.



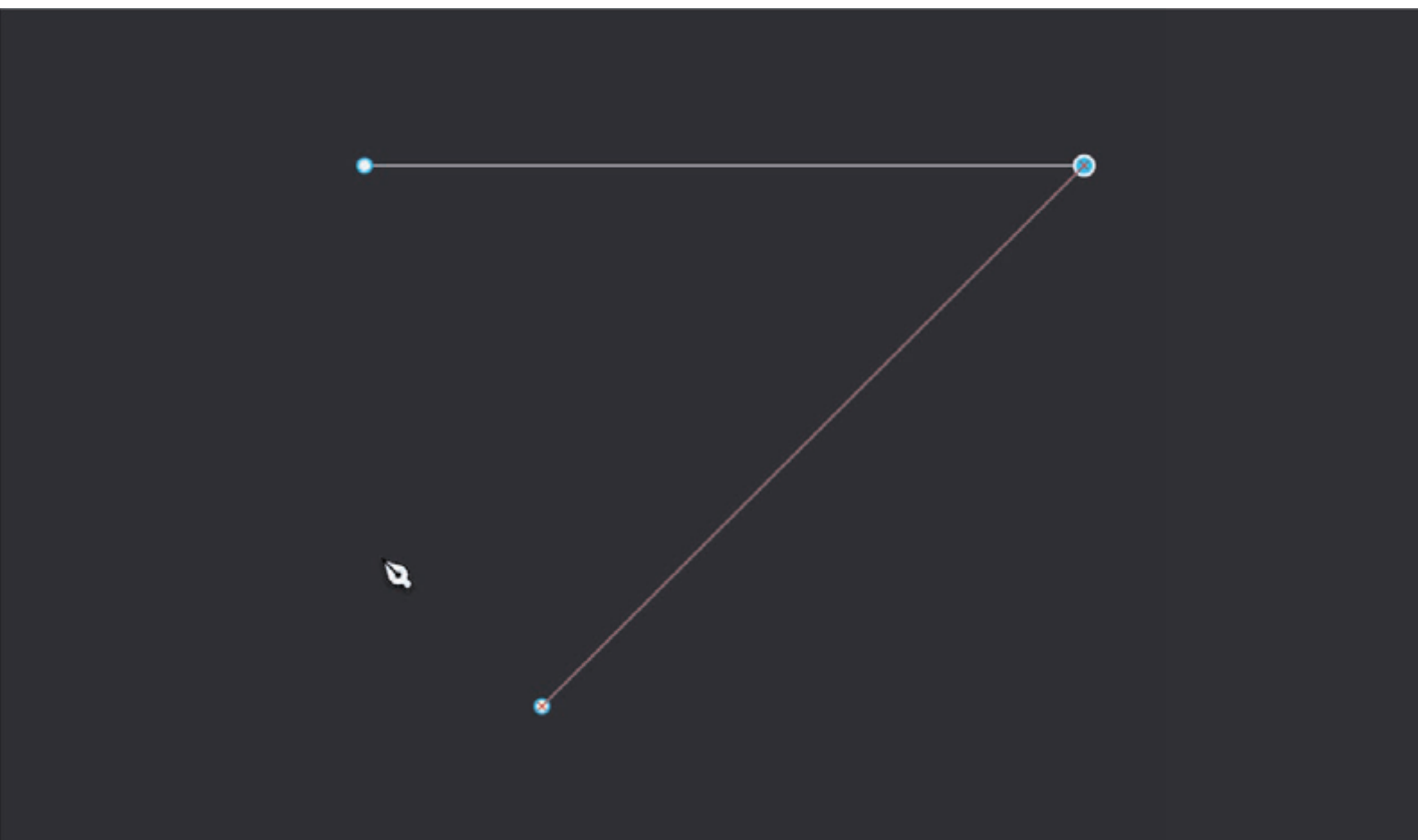
3. Не отпуская **Shift**, кликаем, чтобы зафиксировать линию. Получаем новый шлейф от второй опорной точки.



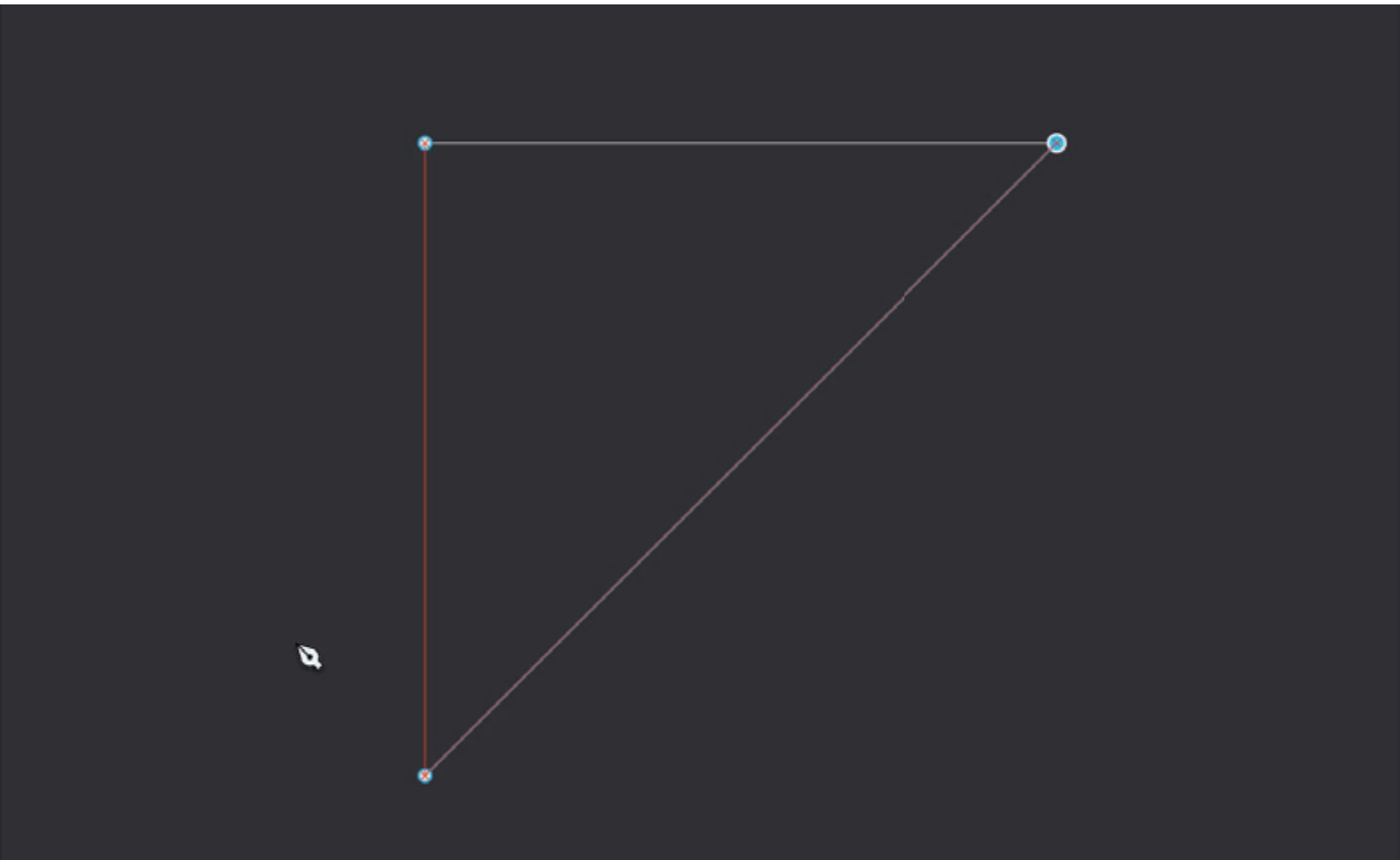
4. Шлейф с зажатим **Shift** может быть направлен в одну из 8 сторон. Помимо горизонтальных и вертикальных линий, можно чертить линии под углом 45° .

Понимание этой концепции здорово повышает точность черчения линий и даже градиентов.

5. Тянем шлейф на юго-запад до момента, пока указатель не будет под первой опорной точкой.

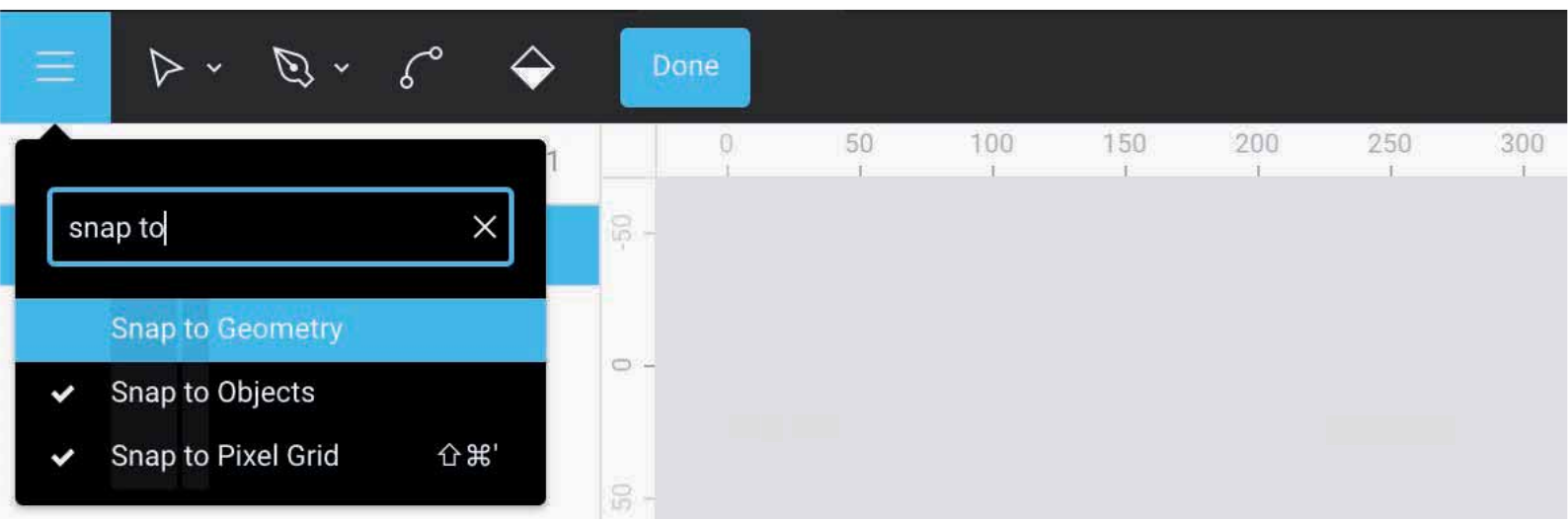


6. Должна появиться красная информирующая линия, подсказывающая, что если мы зафиксируем рисуемую линию, третья точка встанет строго под первой. В Скетче такие линии назывались смарт-гайдами. Ставим третью опорную точку.

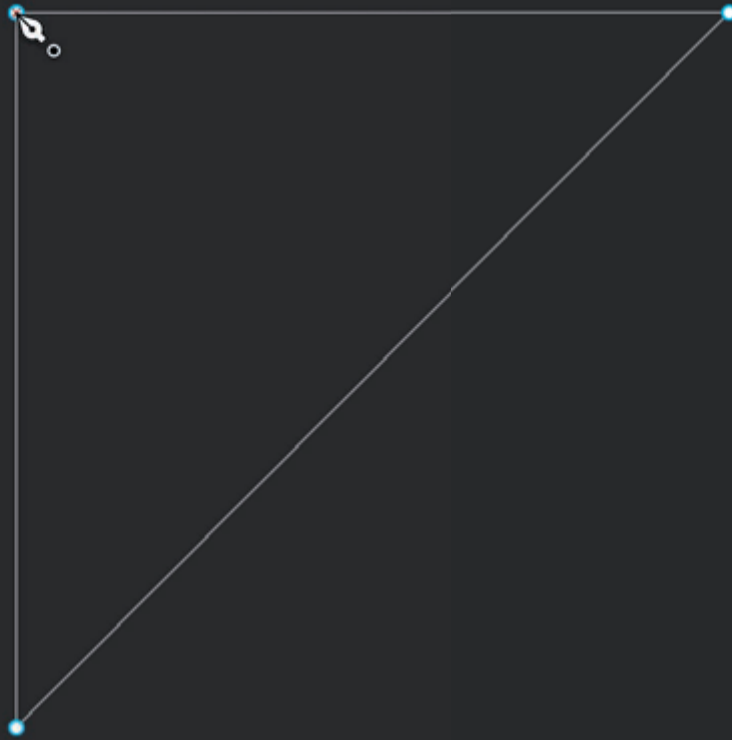


Это сработает, если активен режим **Snap to Geometry**.

Он позволяет липнуть к вспомогательным красным линиям.



7. Тянем последнюю линию от третьей точки до первой и кликаем в первую, закрывая треугольник. Фиксируем фигуру, **Enter**.





10. Векторные сети

[Проект в Фигме →](#)

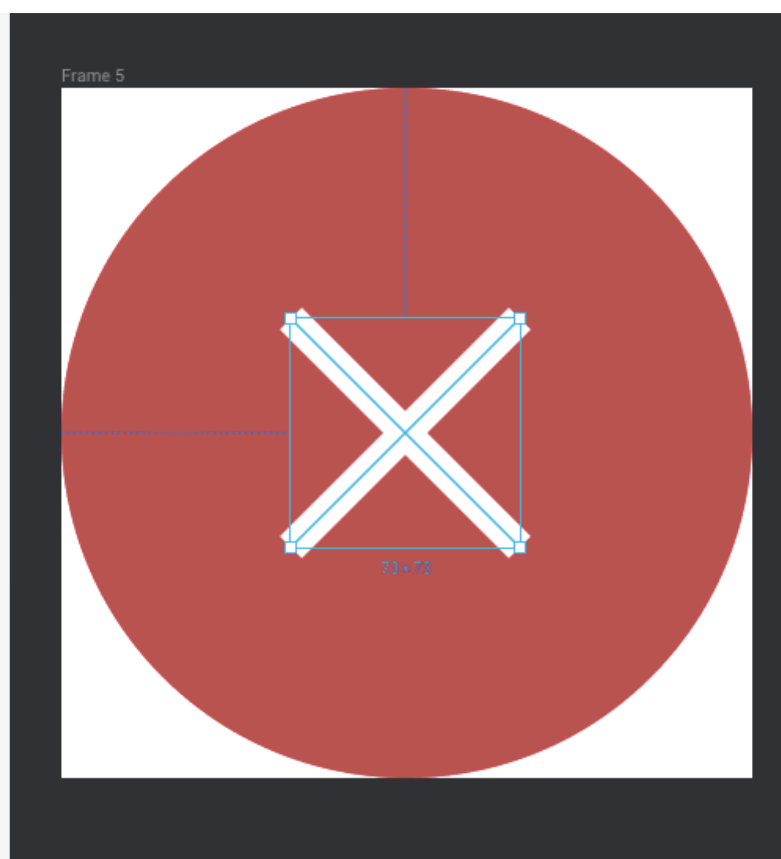
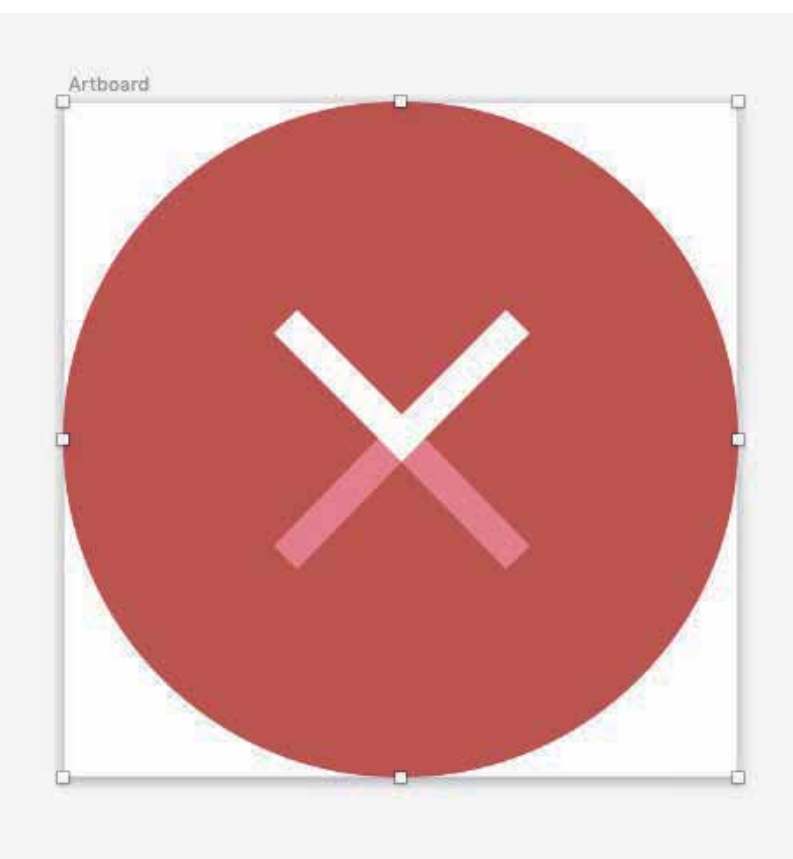
Векторная сеть — это однослойная фигура, состоящая из векторных точек и линий.

В большинстве инструментов, включая Скетч и Иллюстратор, каждая векторная точка может иметь максимум два отрезка, которые к ней прилегают.

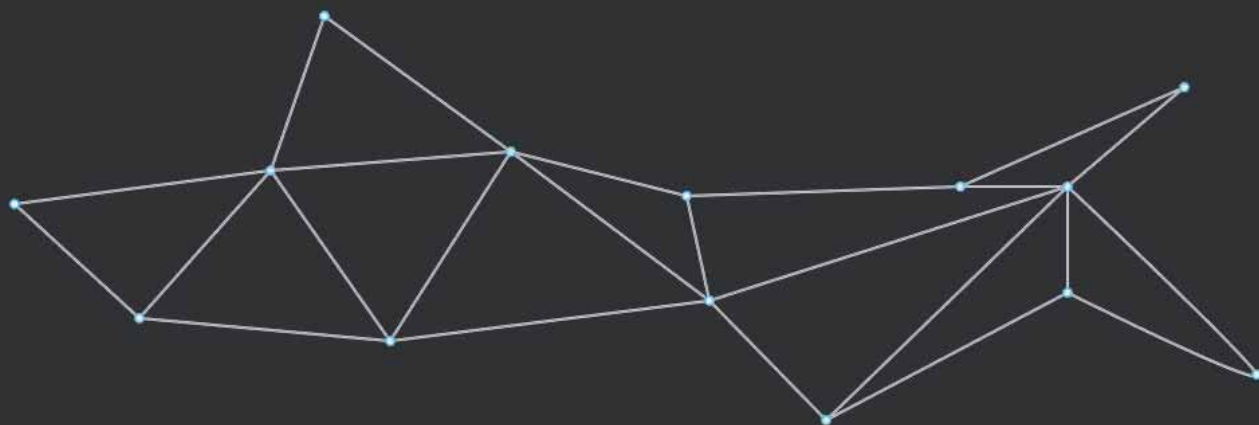
Важно: в Фигме в векторную точку может сходиться любое количество линий. В этом кроются большие творческие возможности.

Например, не нужно накладывать две точки разных объектов друг на друга, чтобы имитировать один объект.

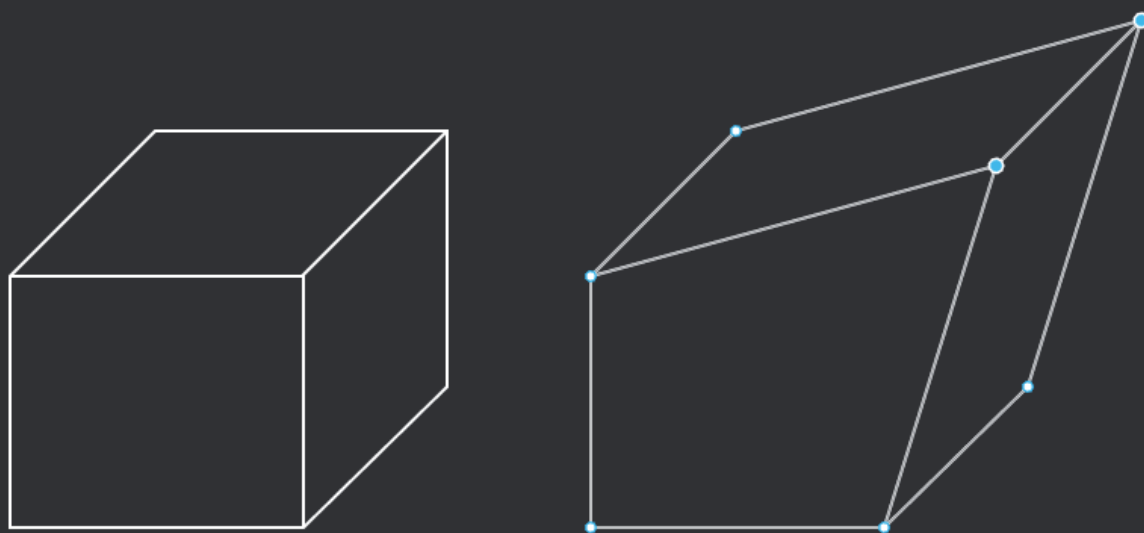
Слева Скetch, справа Фигма. Для наглядности приглушил нижнюю часть креста в Скетче:



Эта возможность позволяет быстро создавать сложные фигуры, которые легко редактировать, не шаманя со съехавшими углами. Пример:

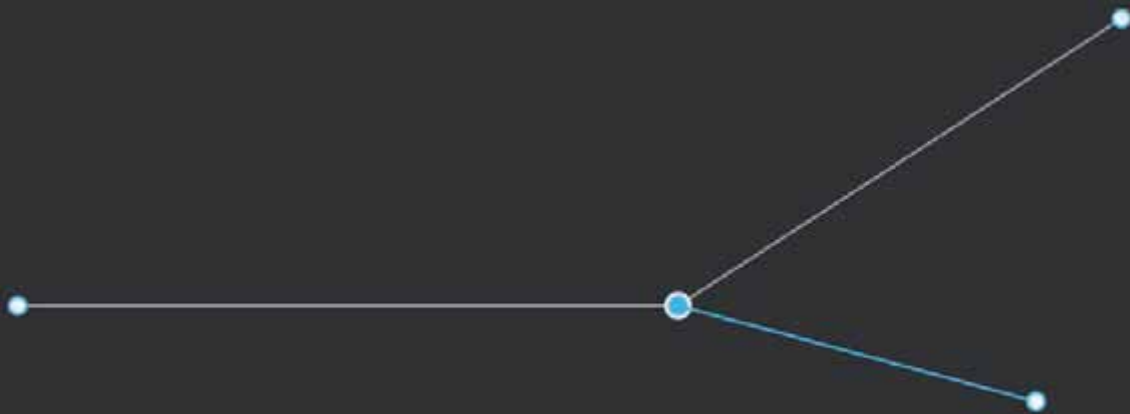


Все точки можно двигать. Толщину всех линий настраивать в один клик. Вместе с точкой передвинутся и все линии, которые к ней тянутся. Выделим две точки, потянем их и фигура перестроится:



Как сделать векторную сеть

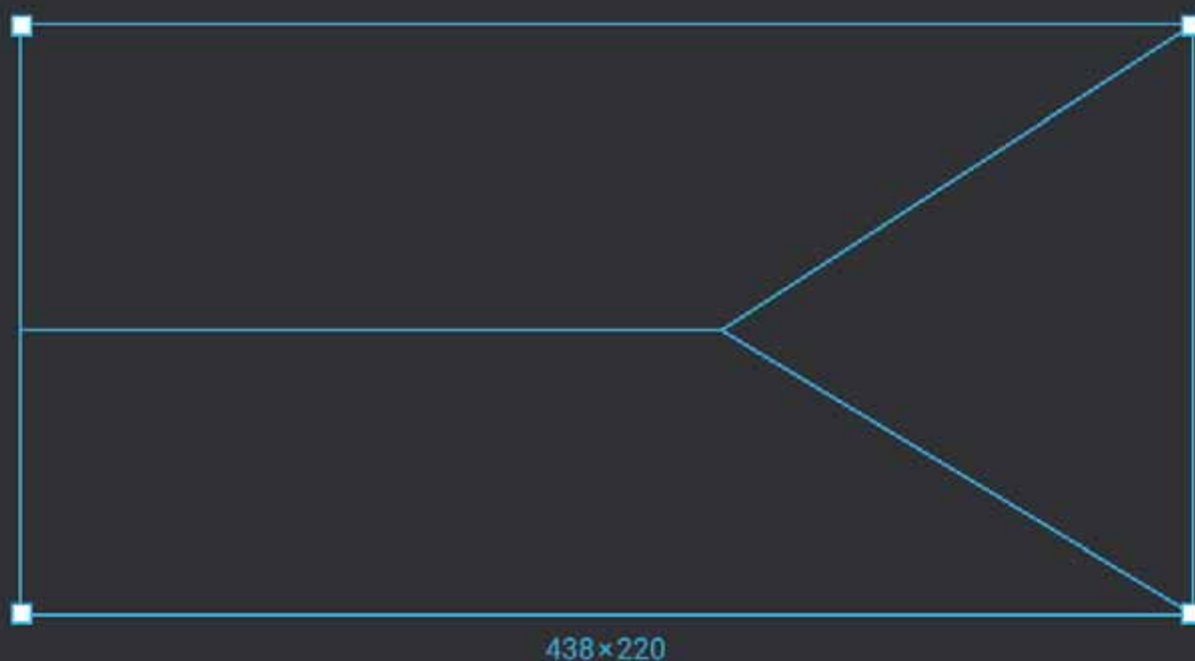
1. Выбираем **перо**, **P**. Рисуем два отрезка. Кликаем в центральную точку между отрезками.



2. Шлейф продолжает тянуться из этой точки.

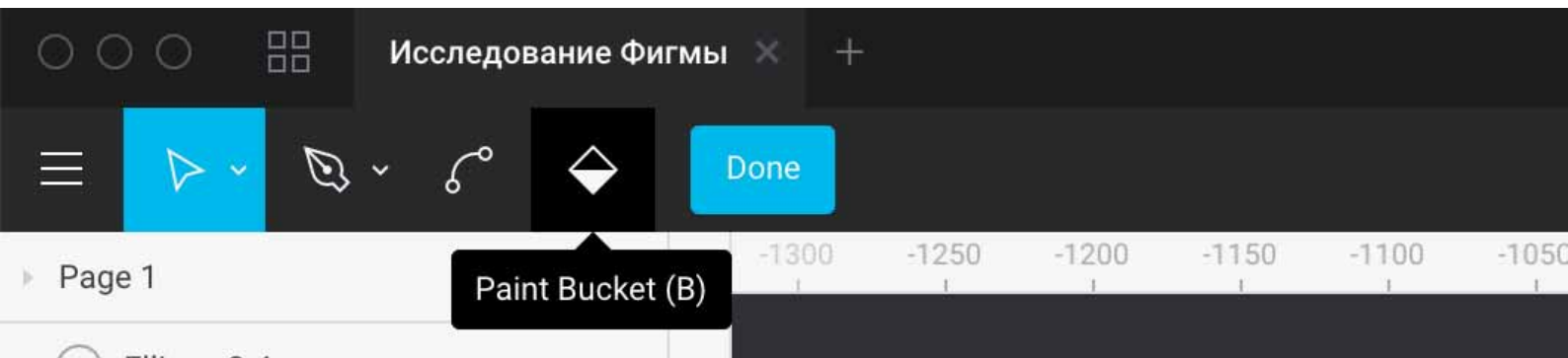


Получилась рогатка, в которой вторая опорная точка имеет три входящие в неё линии. Фиксируем, **Enter**. У нас получилась простейшая векторная сеть. Кажалось бы, пустяк, а именно в этой идее кроется прорыв при работе с вектором.



Заливаем фрагменты сети

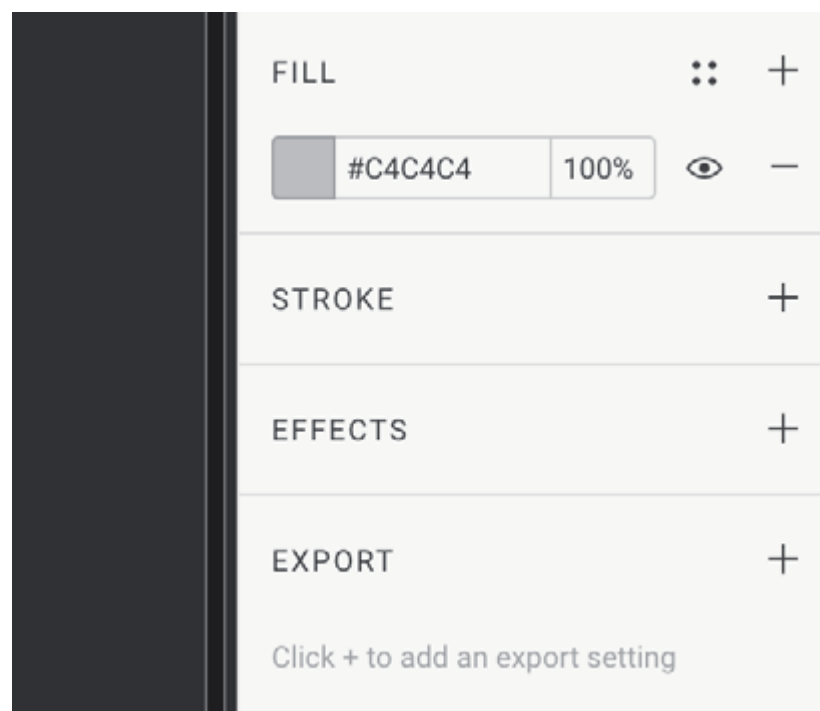
У векторных сетей можно заливать внутренние площади. Для этого используем хорошо знакомый инструмент **Paint Bucket**, **B**, доступный в режиме редактирования. В народе его называют ведёрком.

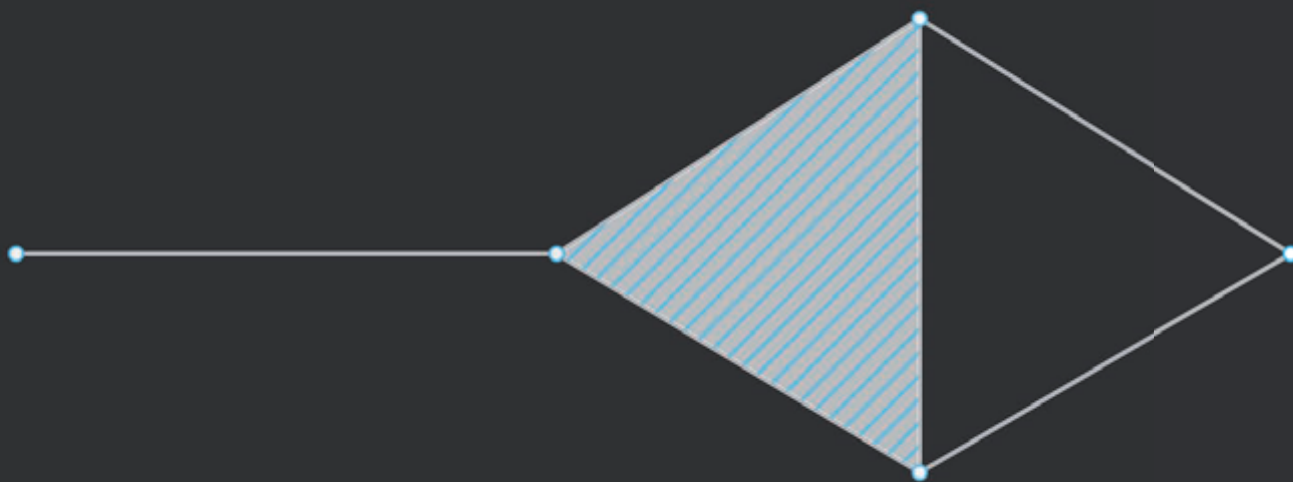


Он позволяет задать те фрагменты фигуры, которые будут подвержены заливке **Fill**. Остальные останутся пустыми.

Если в режиме редактирования выбрать ведёрко, и кликнуть на закрытый фрагмент, он будет залит. Если кликнуть на уже залитый, заливка на нём происходить не будет.

Если же делать заливку через окошко **Fill** в панели справа, залиты будут все фрагменты сети.



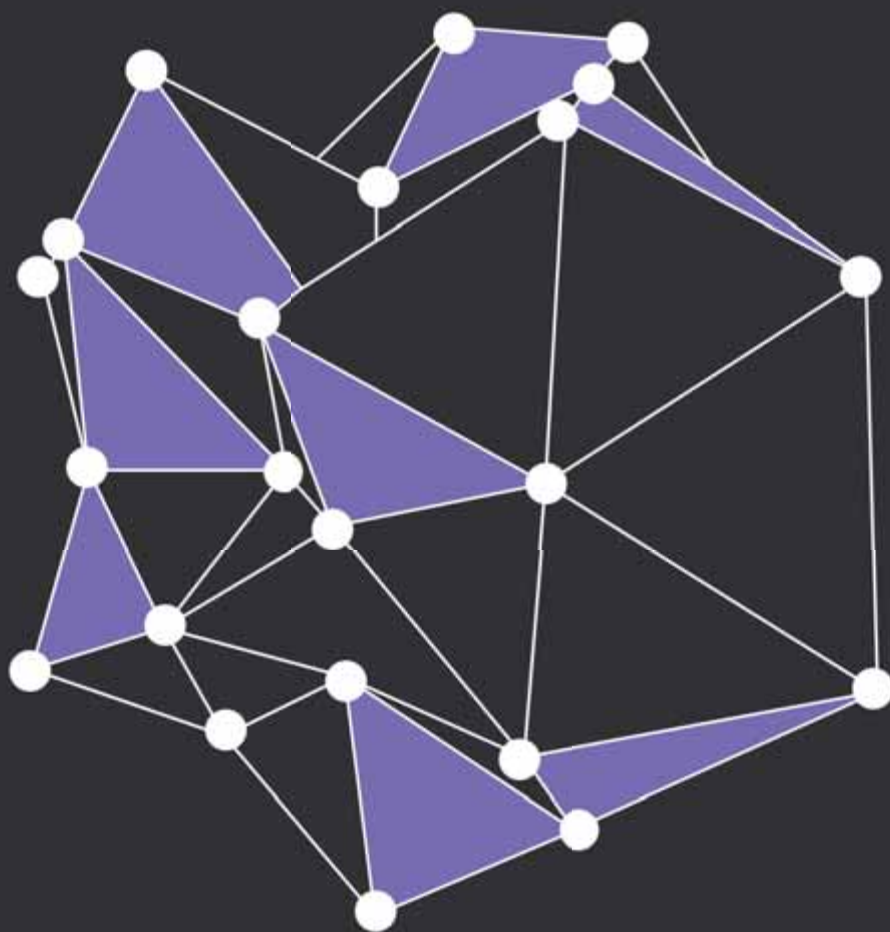


Ограничение: поскольку векторная сеть — один объект, цвет заливки разных треугольников будет единым. Нельзя в рамках работы с одной фигурой один треугольник залить одним цветом, а другой другим.

Bend Tool, **Cmd**

Другой интересный режим работы пера — **Bend Tool**. Он работает, если зажать **Cmd** и позволяет гнуть любые векторные линии, перетаскивая их за центр. Это очень эффективно, потому что не нужно настраивать усы.

Для рисования такой картинki нужно не более 5 минут.



За основу взял 3D-рендер из группы [BLGN VFX](#)

Рисуем иконки

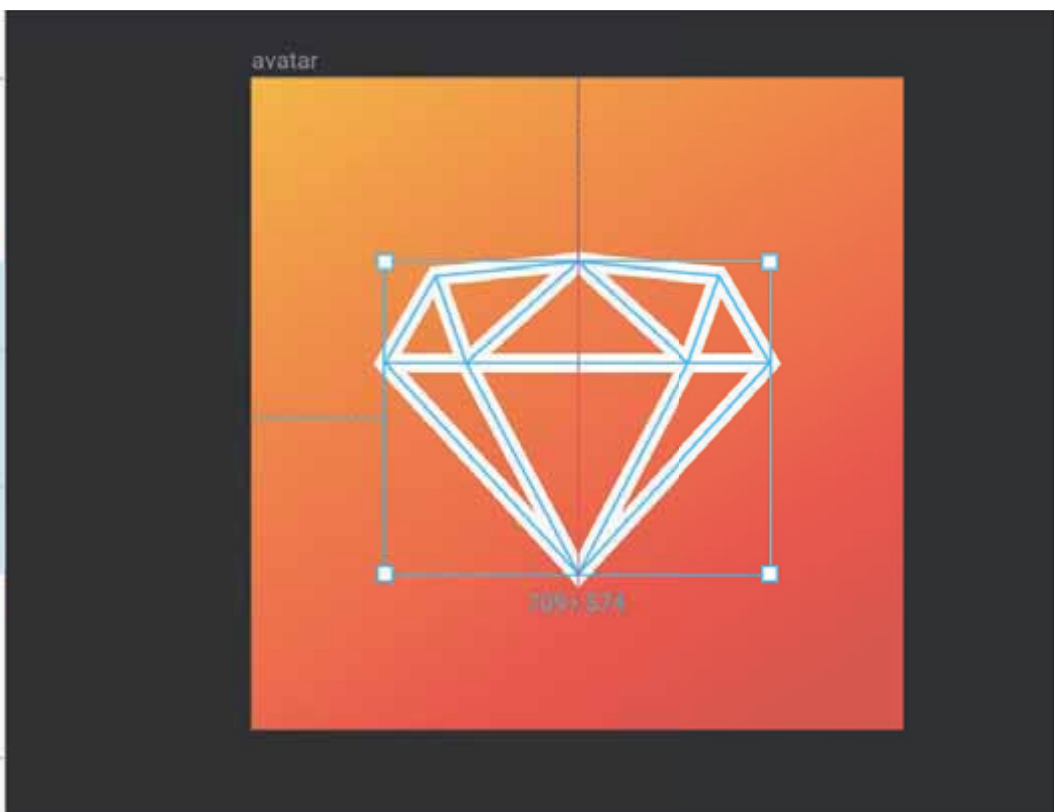
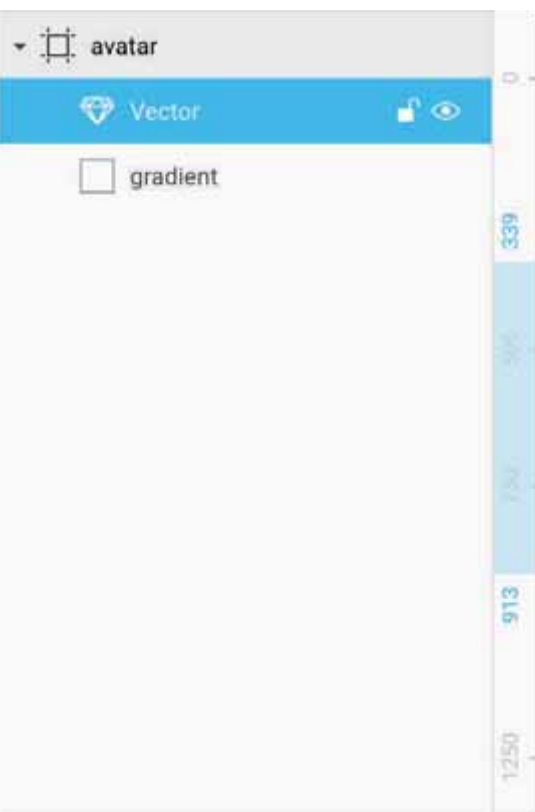
Векторные сети дают большую гибкость в работе с толщиной линии. Например, чтобы нарисовать аватар моего канала в Скетче, мне пришлось использовать несколько линий.



Если изменить толщину линии, иконка рассыпется:



Если нарисовать ту же фигуру в Фигме, толщину линий можно менять как угодно.





11. Булевы группы и флэттен

[Проект в Фигме →](#)

Если нужна более сложная форма, чем стандартные прямоугольники, круги или треугольники, её можно нарисовать при помощи пера. Однако, если сложную форму можно разбить на несколько базовых геометрических фигур, будет быстрее комбинировать их в одну.

Булевыми группами в Фигме называются принципы взаимодействия шейпов друг с другом.

В Скетче они называются **булевыми операциями**.

Пример, в котором перо будет вовсе бесполезно: если нужно нарисовать пончик. Мы можем составить его из двух кругов, наложив синий на розовый:



Возникает проблема: если поменяется фоновый цвет, внутренний круг останется старого цвета. Чтобы в центре розового круга образовалось отверстие, нужно вычесть из него форму синего. Для этого мы заключаем оба круга в булеву группу и ставим её в режим вычитания.

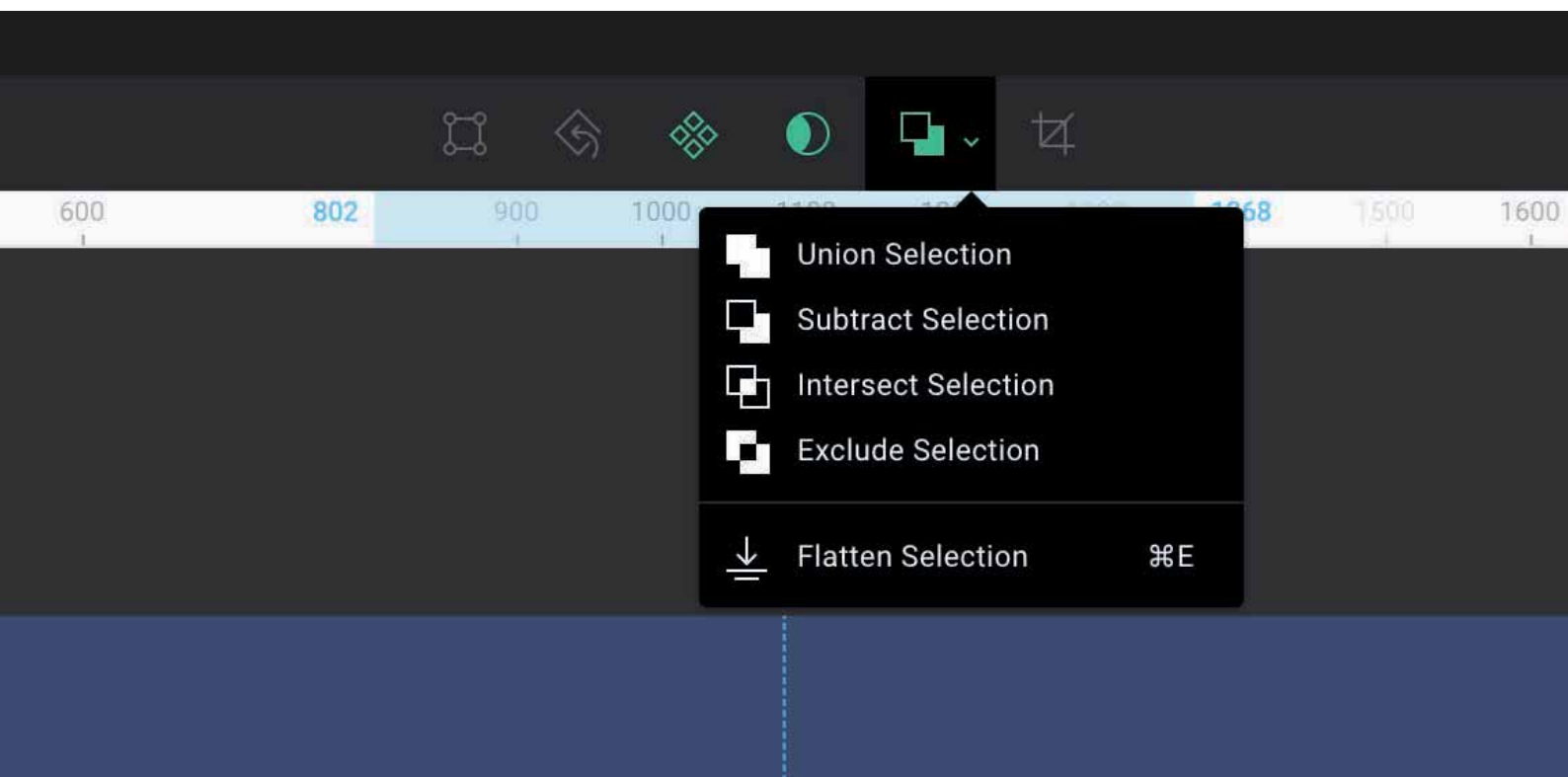
Булево значение может быть в двух режимах: **true** и **false**.

Слово *булевый* здесь означает, что шейп может быть либо видимым (true), либо невидимым (false). В зависимости от режима, шейп ведёт себя определённым образом.

Всего бывает 4 режима булевых групп:

- **Union** — Объединение двух фигур в одну. Видимы обе.
- **Subtract** — Вычитание верхней из нижней. Видима нижняя.
- **Intersect** — Видима область пересечения двух фигур.
- **Exclude** — Видимы обе фигуры, не видима область пересечения.

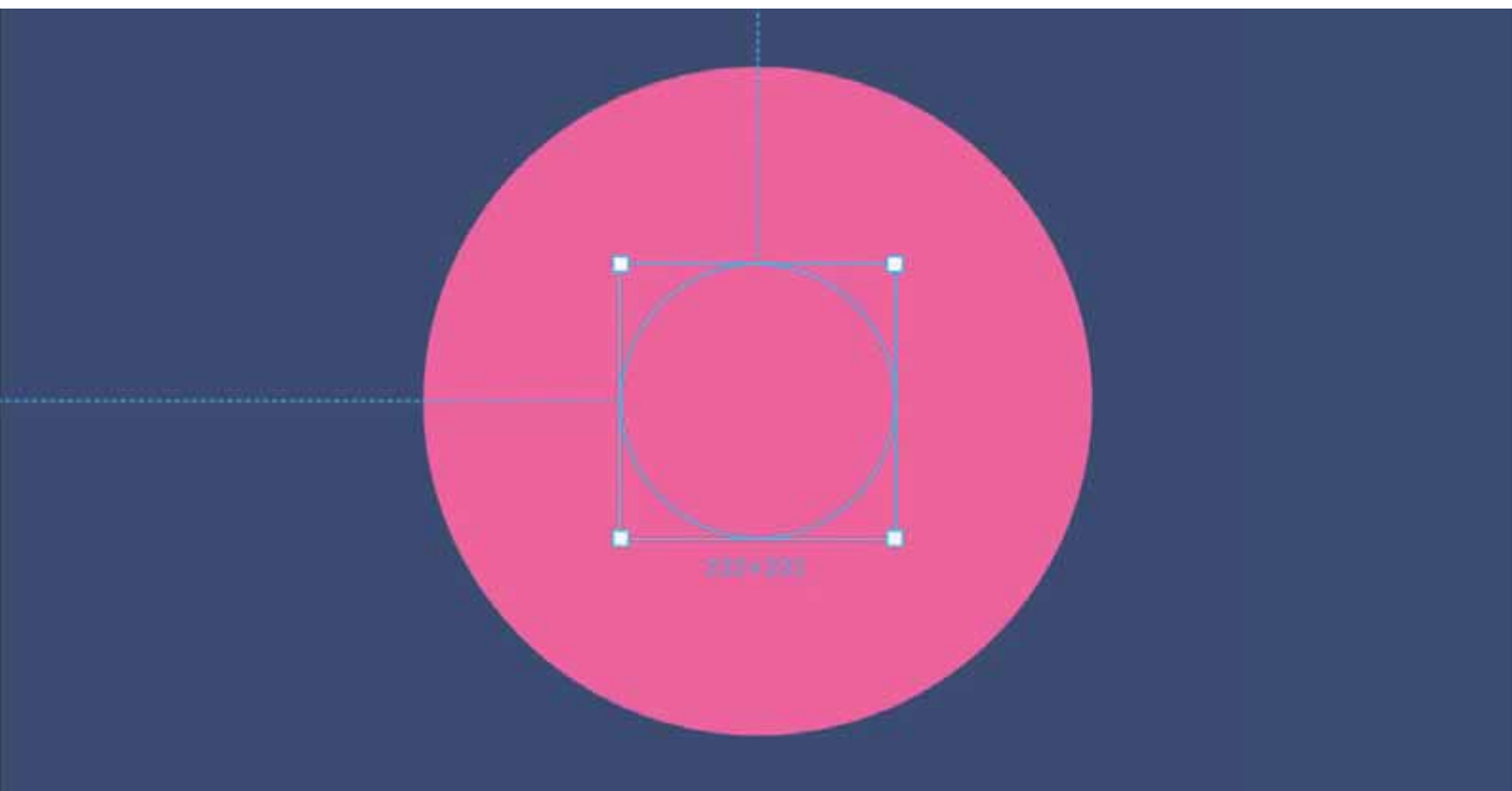
Задав один режим, можно переключить группу на другой или разгруппировать её, **Shift + Cmd + G**.



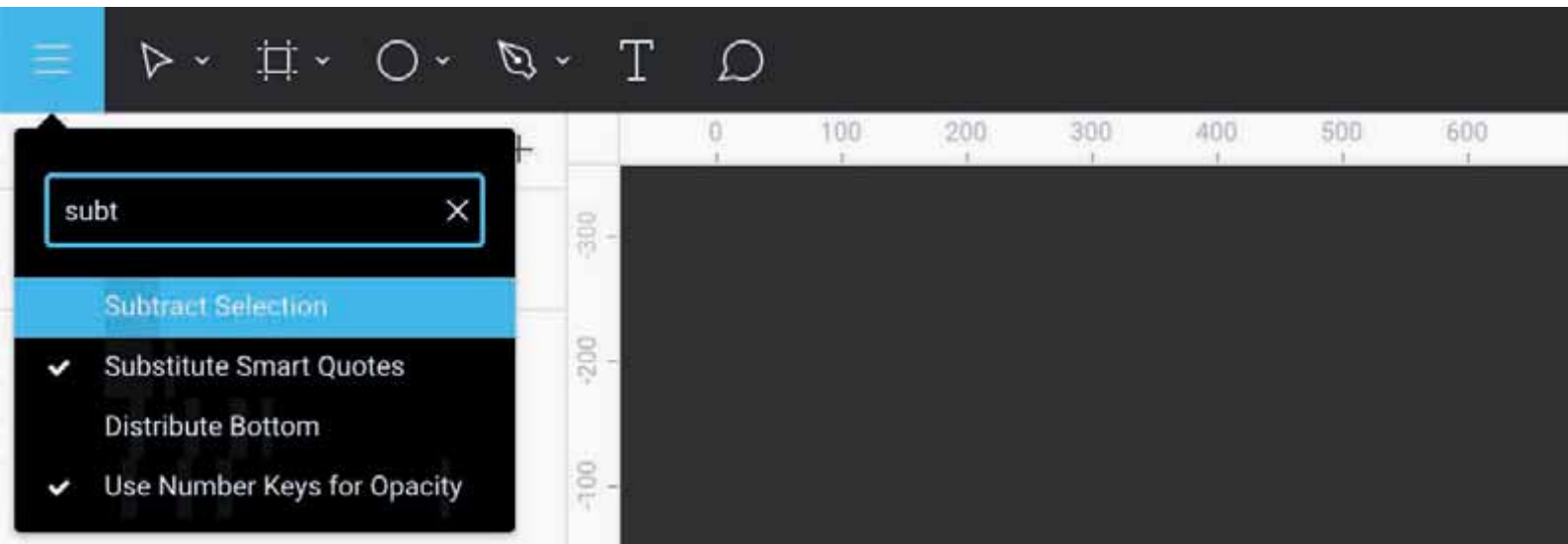
Subtract на практике: вычитаем круги

Простейший пример с пончиком.

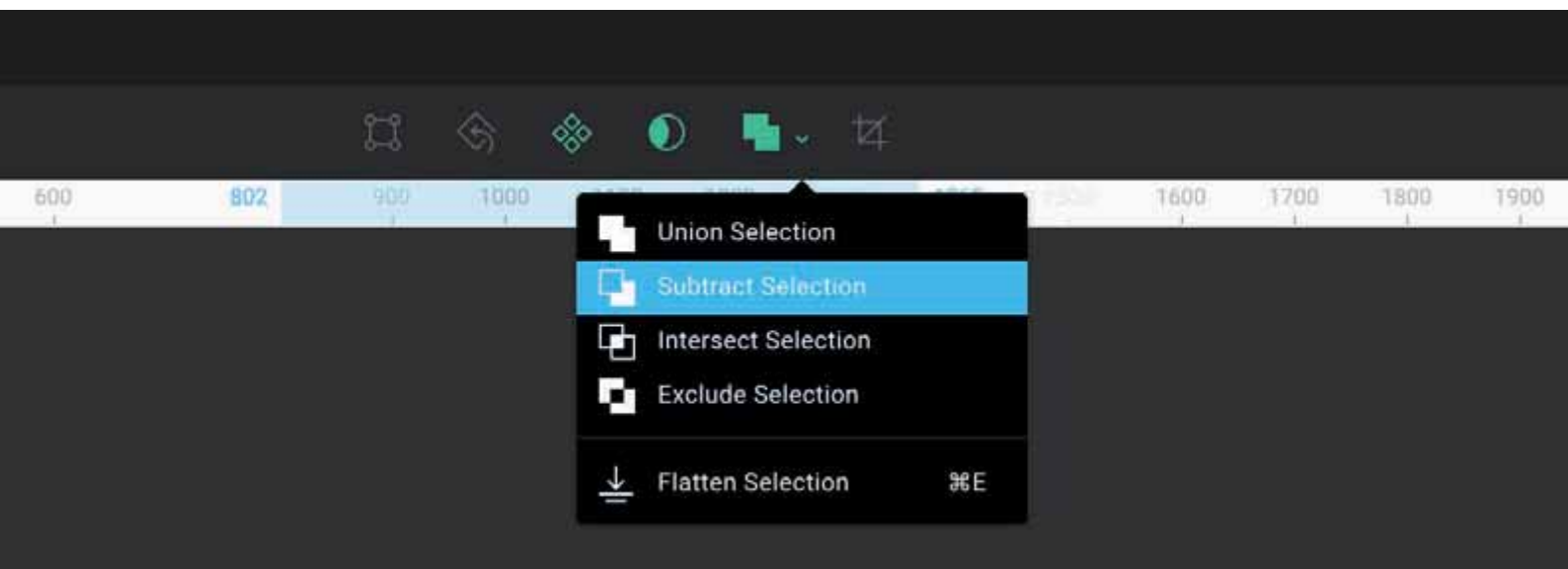
1. Рисуем круг, **O**. При желании заливаем его цветом в поле **Fill**.
2. Дублируем его клавишей **Cmd + D**. Теперь два одинаковых круга лежат стопкой.
3. Зажимаем нижнюю правую квадратную ручку, чтобы уменьшить верхний круг. Тянем вверх и влево, к его центру.
4. Пока тянем к центру, большим пальцем зажимаем **Opt**, чтобы опорой для масштабирования круга была не левая верхняя точка, а геометрический центр.
5. К зажатому **Opt** добавляем **Shift** мизинцем. Он фиксирует пропорции круга, не позволяя нам рисовать эллипс.
6. Уменьшаем круг до нужного размера.



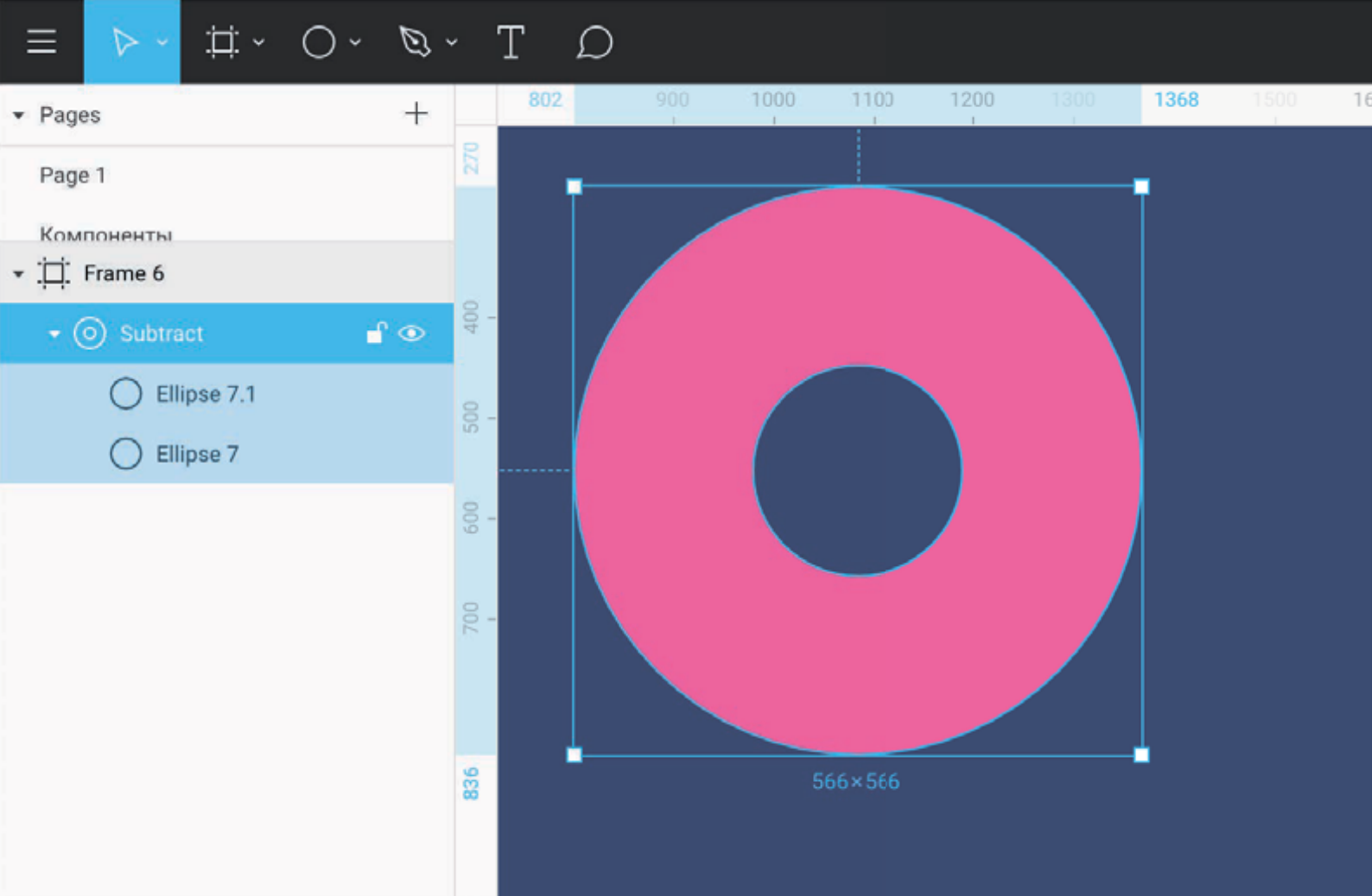
7. Выделяем оба слоя, зажав **Shift** и последовательно кликнув их.
8. Применяем команду **Subtract Selection**, которая доступна в тулбаре:



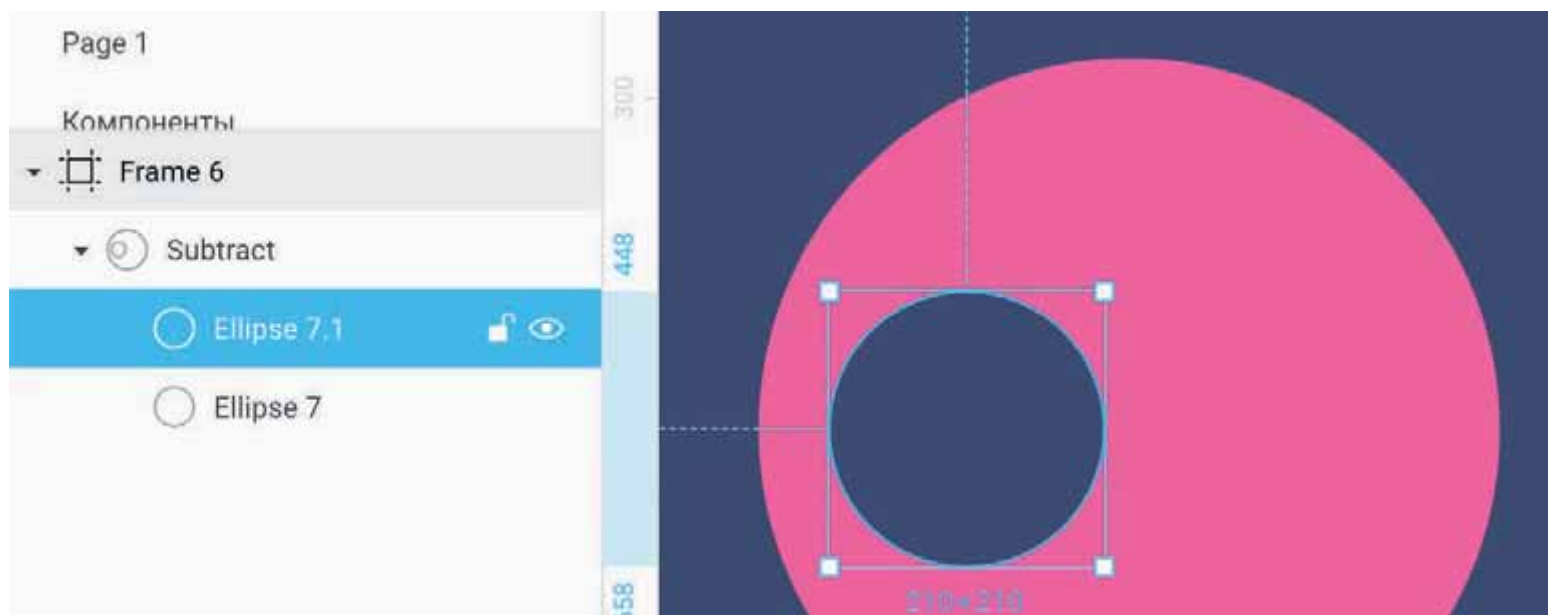
Также её можно вызвать с клавиатуры через поиск: **Cmd + /**, **subt**.



9. В результате получается булева группа с названием **Subtract**, в которой соединены оба круга. Верхний вычитается из нижнего.



Каждый из кругов всё ещё можно редактировать внутри группы. Также группу можно разгруппировать, **Shift + Cmd + G**.



Union на практике: соединяем шейпы

Пример с молотом, аналогичный **Subtract**.

1. Рисуем два прямоугольника, **R**, чтобы получилось что-то вроде молота:



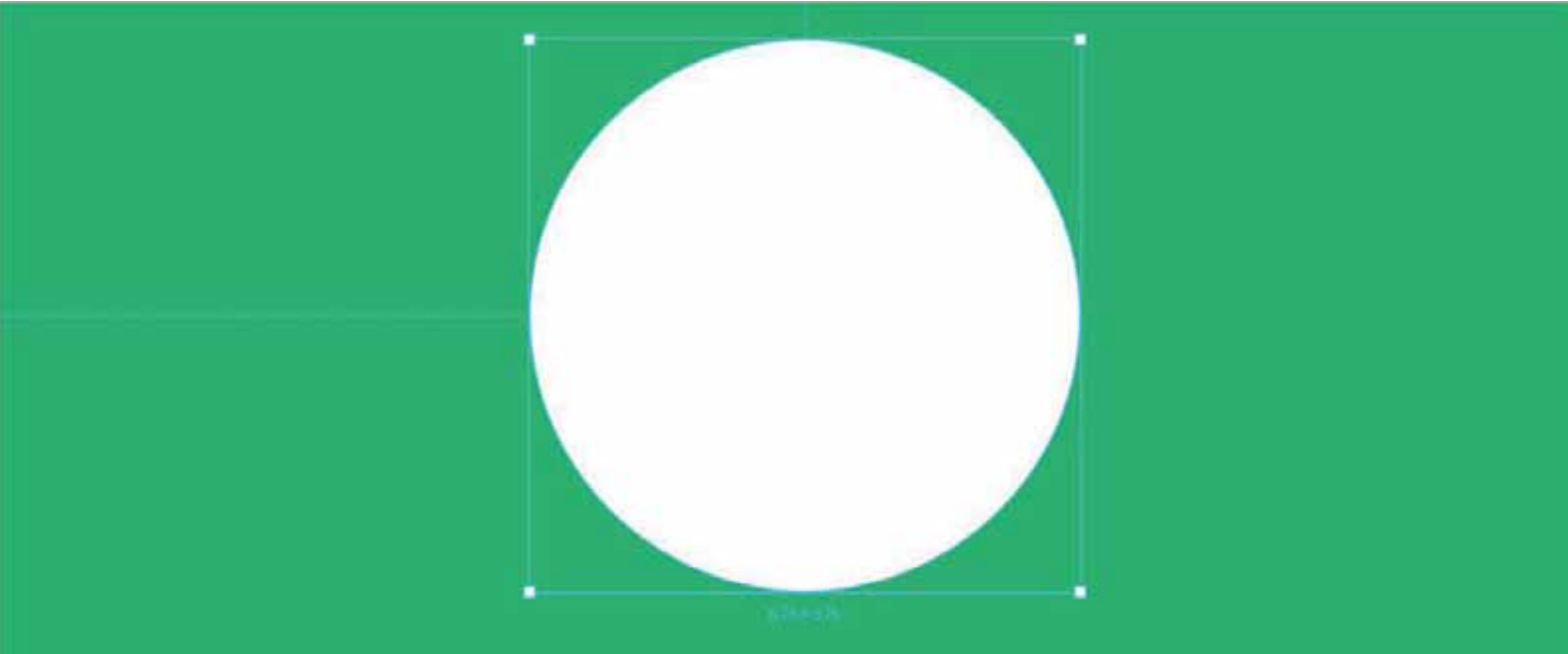
2. Выделяем оба с **Shift**.
3. Применяем команду **Union Selection**, **Cmd + /**, **uni**.



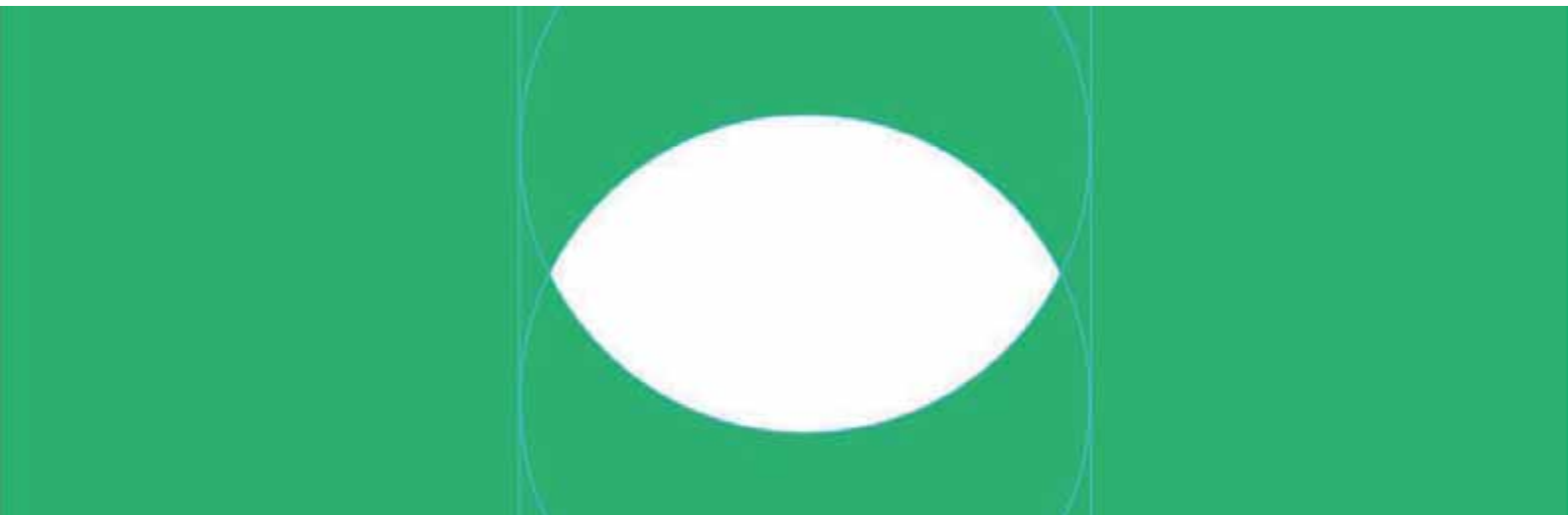
Intersect на практике

Нарисуем мяч для регби. Проявим пересечение двух кругов.

1. Рисуем круг, **O**.



2. Дублируем его, зажав **Opt**. Тянем вниз.
3. Выделяем оба, применяем **Intersect Selection**, **Cmd + /**, **in**.
4. Результат:



Exclude на практике

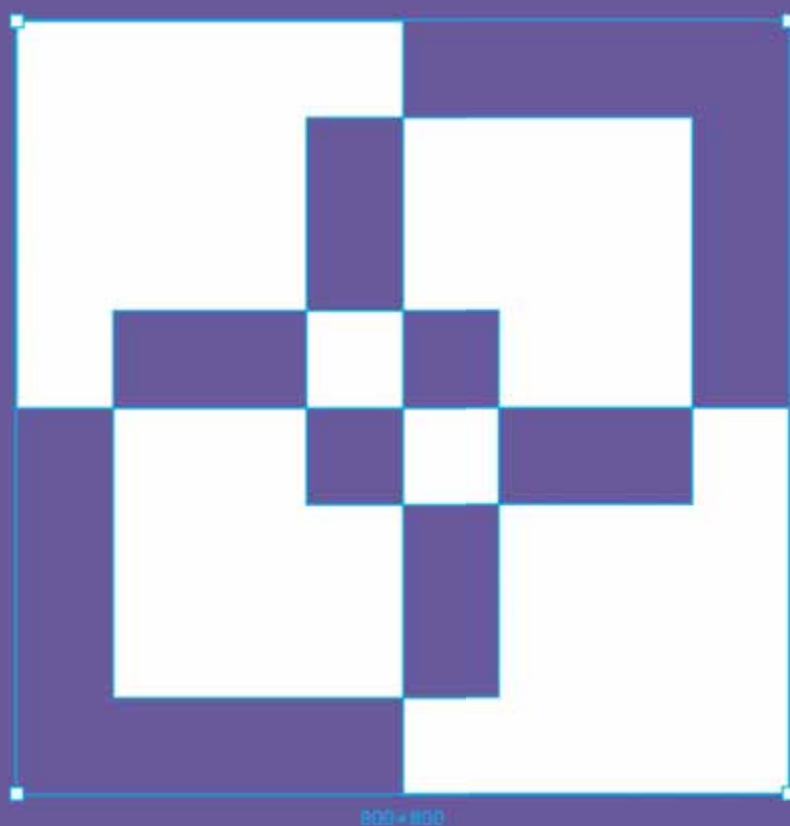
Режим, который в случае с двумя фигурами скрывает общие фрагменты пересекающихся форм.

Нарисуем такую фигуру:



1. Рисуем квадрат, **R**.
2. Зажав **Opt**, вытягиваем его дубль. Ставим угол одного квадрата в центр другого.
3. Выделяем оба, переключаем их в режим **Exclude Selection**, **Cmd + /, ex**.

Exclude — инструмент импровизаторов. Режим позволяет достигать интересных и непредсказуемых результатов. Если добавить к изначальной фигуре из двух квадратов ещё два и смешать их, получится более сложный геометрический узор:



Нарисовать такую геометрическую наркоманию в стиле 70-х за несколько минут без **Exclude** было бы невозможно.



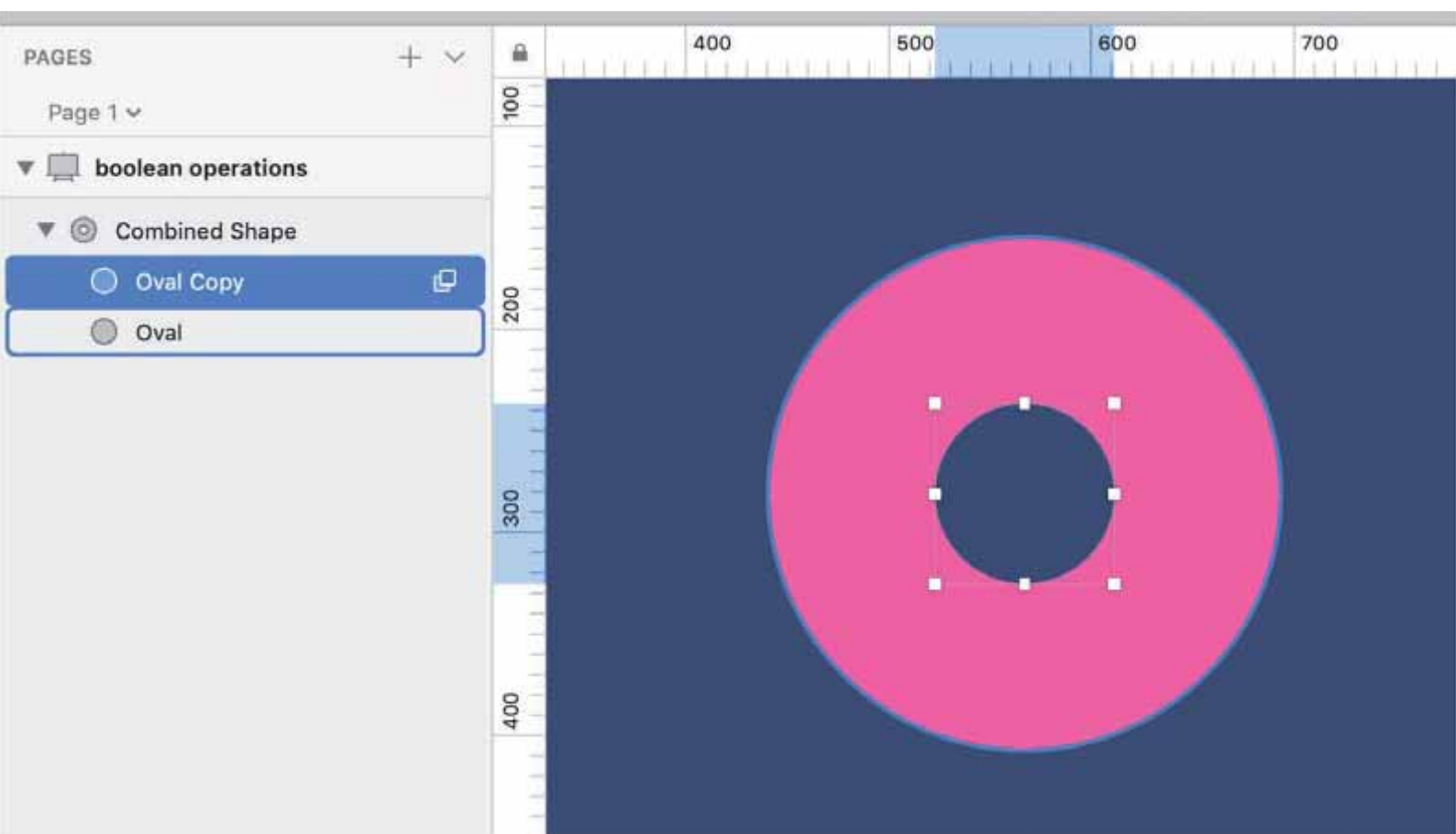
Забегая вперёд в тему текстовых слоёв: булевы режимы работают и с текстом. Два слоя в режиме **Exclude** и с небольшим смещением на геометрическом шрифте **Gilroy ExtraBold** могут давать такой эффект:



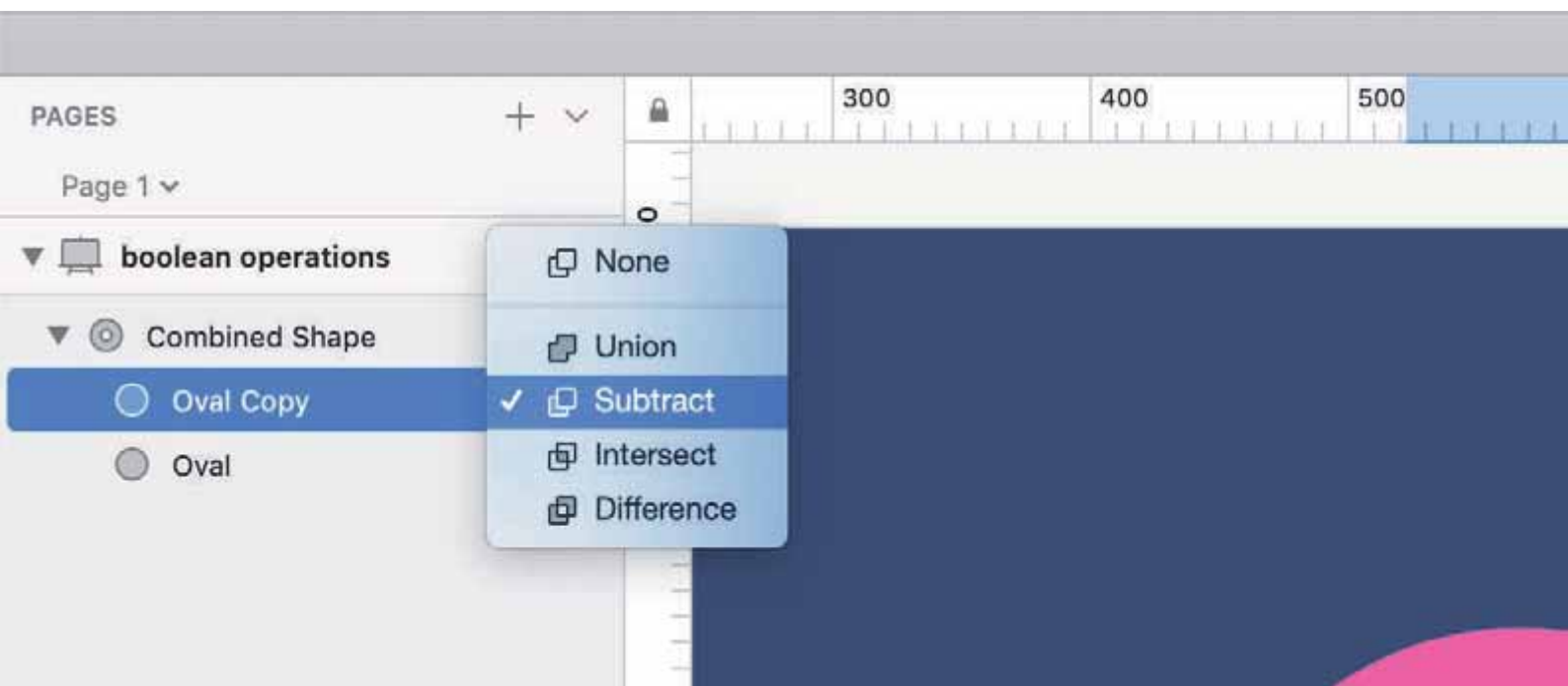
Сходства и различия булевых групп в Фигме и Скетче

Boolean Groups и **Boolean Operations** в целом похожи, однако, есть небольшие различия.

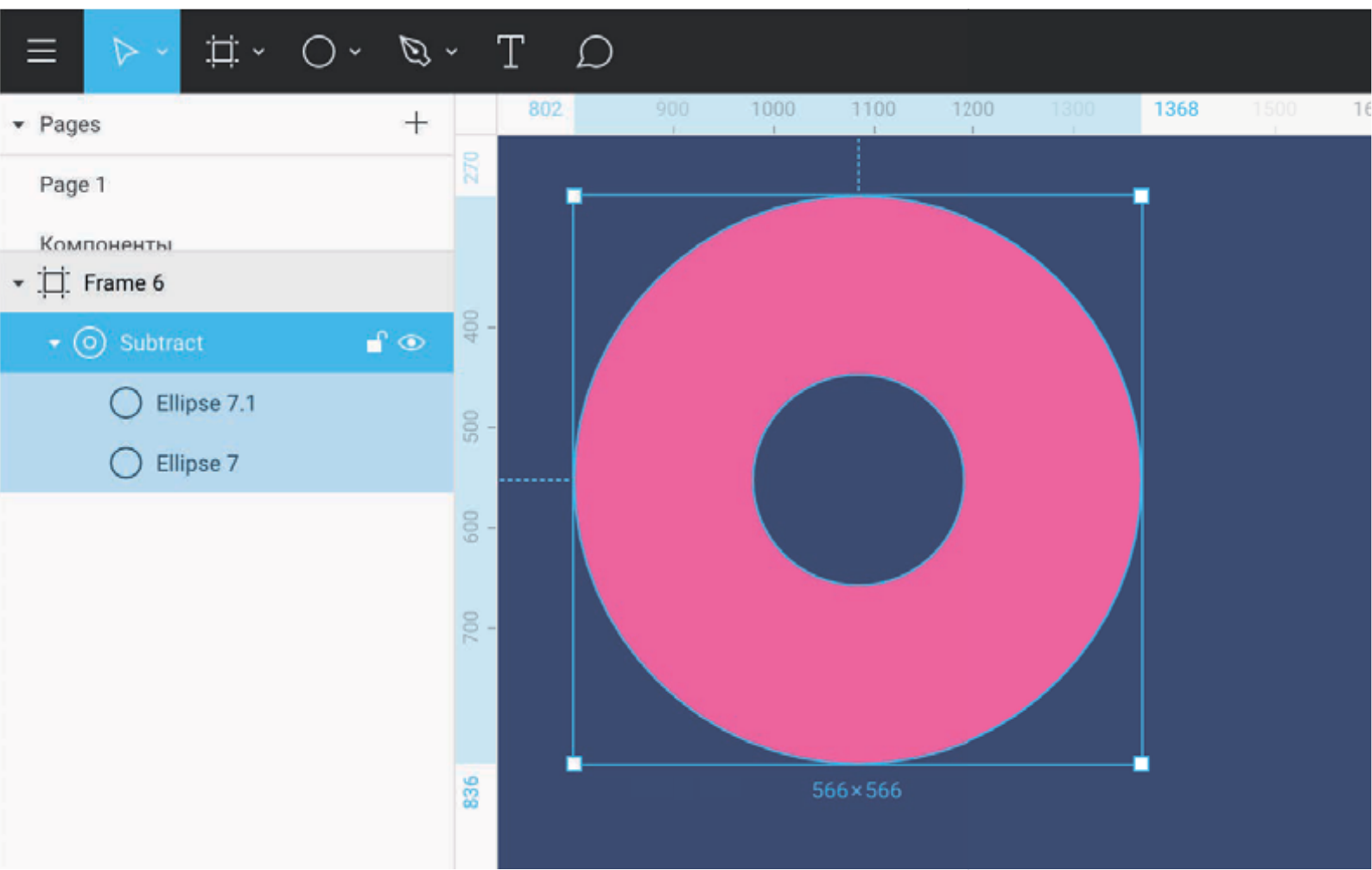
В Скетче булев режим — это настройка шейпа. Два разных шейпа могут быть в разных режимах. Вернёмся к примеру с пончиком и повторим его в Скетче. Есть розовый круг, а из него вычитается меньший круг в режиме **Subtract**. В слое видна его иконка справа.



По клику на неё можно переключать режимы у этого слоя.

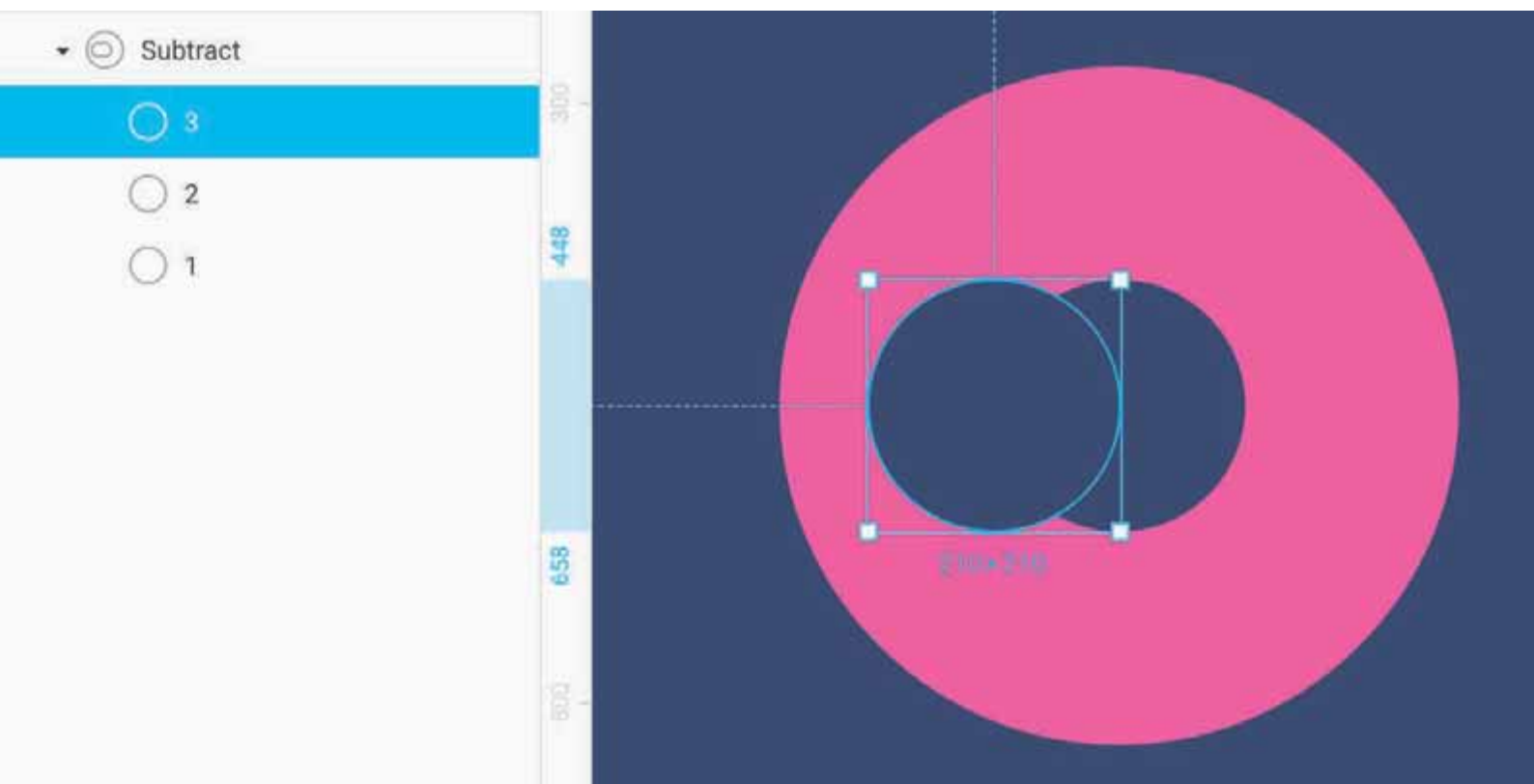


В Фигме же булев режим — это свойство целой группы. Отдельный слой не может быть в режиме **Subtract** или **Union**.



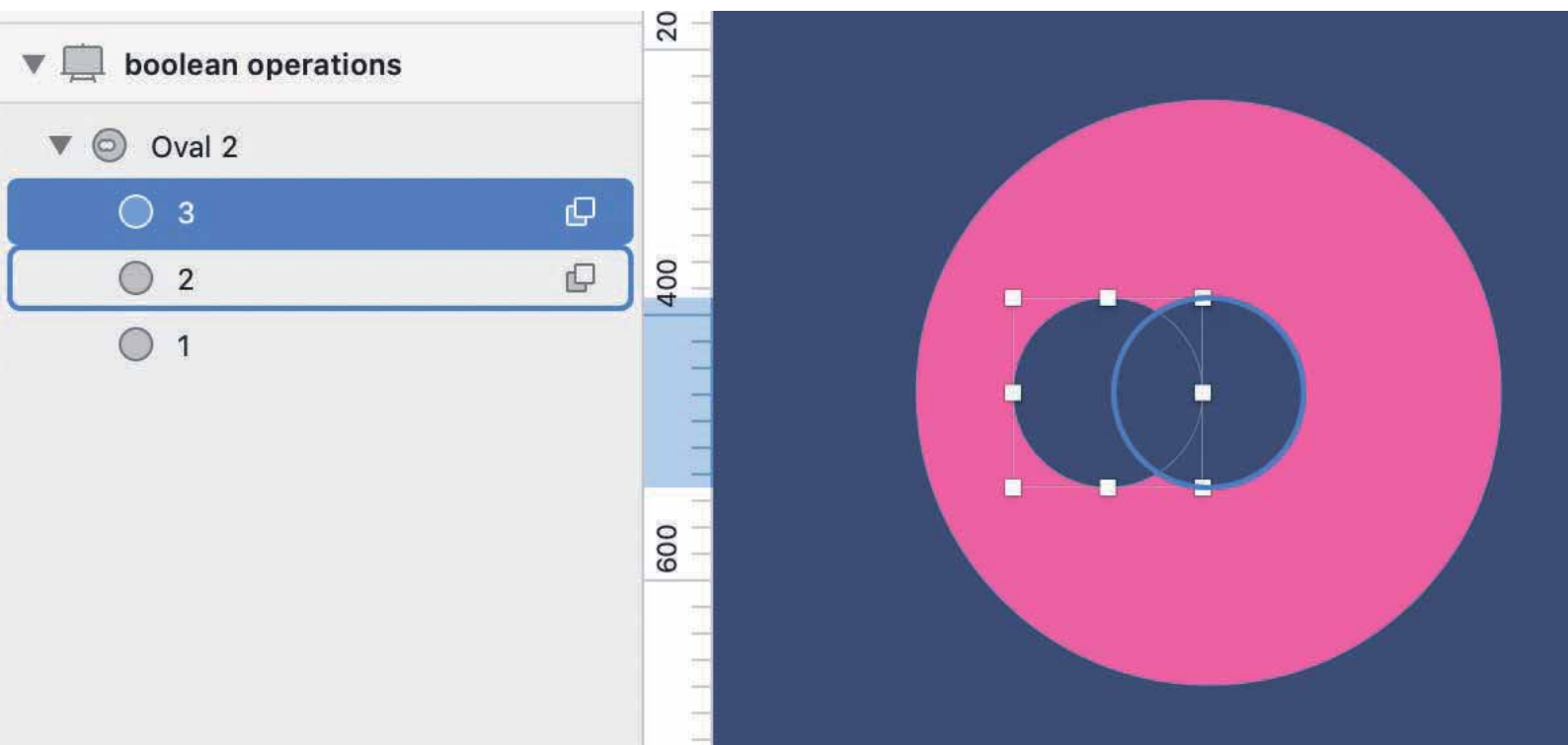
В Скетче и Фигме нижний слой является базовой фигурой, из которой происходит вычитание. Если дублировать вычитающую фигуру и сдвинуть её, она увеличит площадь вычитания.

Фигма:



Как мы видим, если у группы стоит режим **Subtract**, все верхние фигуры внутри этой группы **2** и **3** вычитаются из нижней **1**.

Поскольку мы дублировали слой **3** из слоя **2**, он унаследовал режим **Subtract**. В правой части списка видна его иконка.



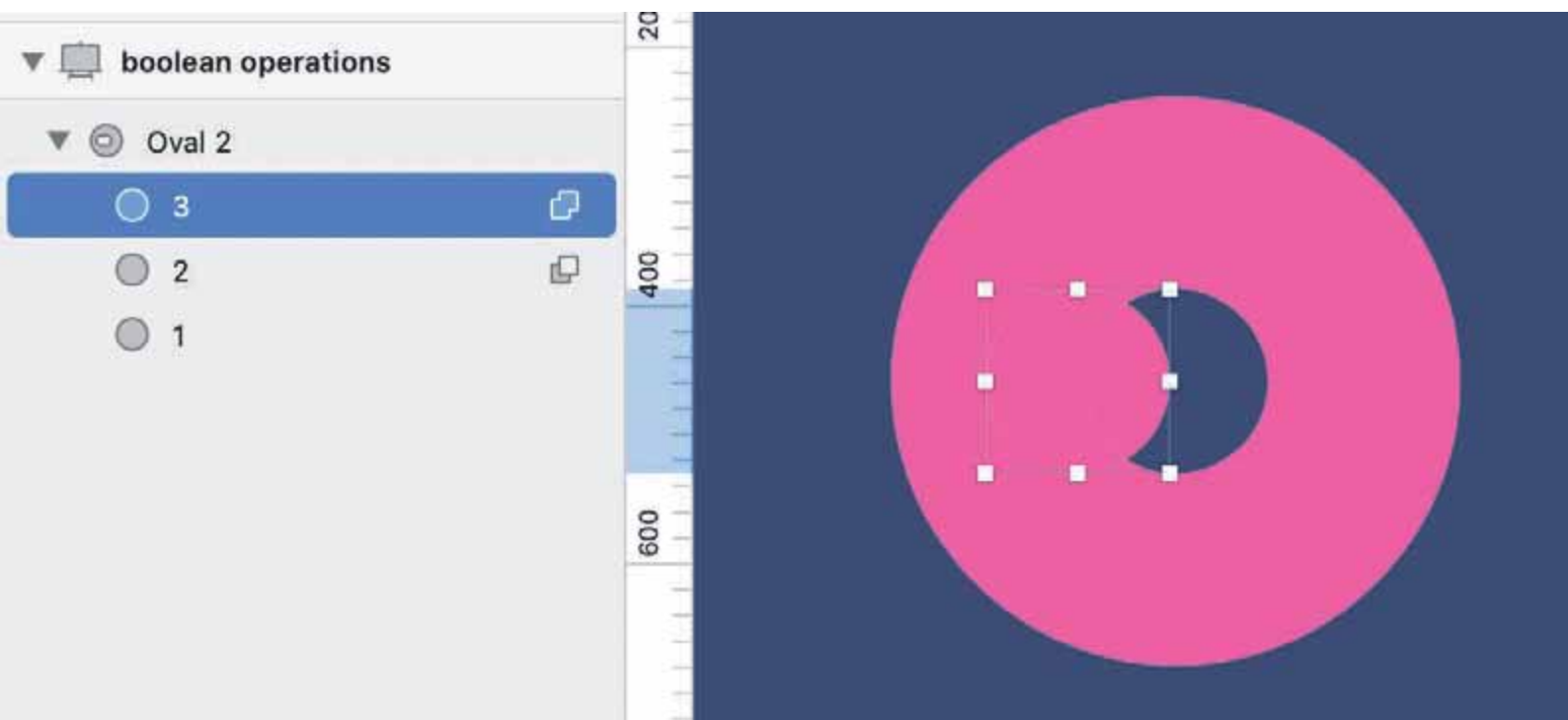
Принцип инверсии в Скетче

Чтобы вычитать из негативного пространства, надо добавлять.

Если нужно, чтобы верхний круг **3** отрезал кусок от среднего круга **2**, находясь внутри нижнего круга **1**, можно переключить круг **3** в режим **Union**, который складывает формы.

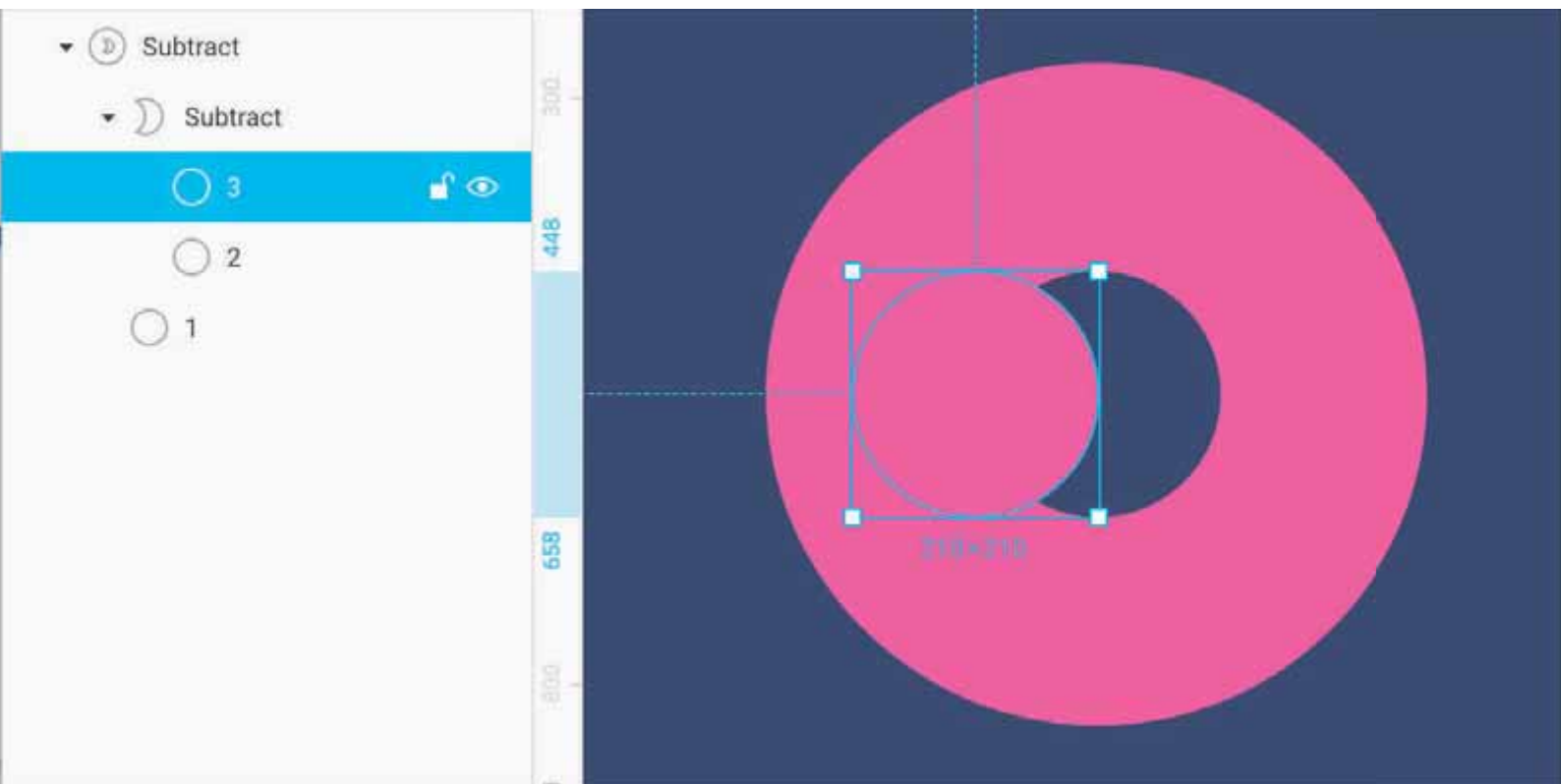
В зависимости от последовательности слоёв, Скетч будет просчитывать итоговую форму, двигаясь от нижнего слоя к верхнему:

1. Из нижнего круга **1** вычитает круг **2**. Получается пончик.
2. К нижнему кругу **1** прибавляет свою форму круг **3** в режиме **Union**. Круг 3 оставляет от круга **2** форму полумесяца:



Вложенные булевы группы в Фигме

Поскольку в Фигме нельзя отдельному слою задать булев режим, нужно сделать вложенную булеву группу. Для этого внутри группы **Subtract** нужно выделить слои **2** и **3** и снова нажать **Subtract**.



Теперь базовым слоем внутри этой группы стал слой **2**, а из него вычитается слой **3**. Получается полумесяц. Затем, по правилу первой группы **Subtract**, он вычитается из слоя **1**.

Надо признать, что в Скетче переключать режимы в списке слоёв быстрее, чем настраивать вложенные группы в Фигме. В остальном реализация даёт равные возможности.

Flatten составных фигур

Фигуры, которые мы создаём через булевы группы, состоят из простых объектов.

Однако, бывает необходимость соединить многослойную форму в однослойную, когда не хочется отвлекаться на структуру фигуры, а нужно работать с ней как с единым целым. Для этого в Фигме используется функция **Flatten Selection**, и у неё нелогичная стандартная клавиша **Cmd + E**.

Если делаешь **Flatten**, желательно делать копию исходной формы, чтобы всегда иметь возможность разобрать её на составные части, а не мучаться, редактируя плоскую фигуру в кривых.

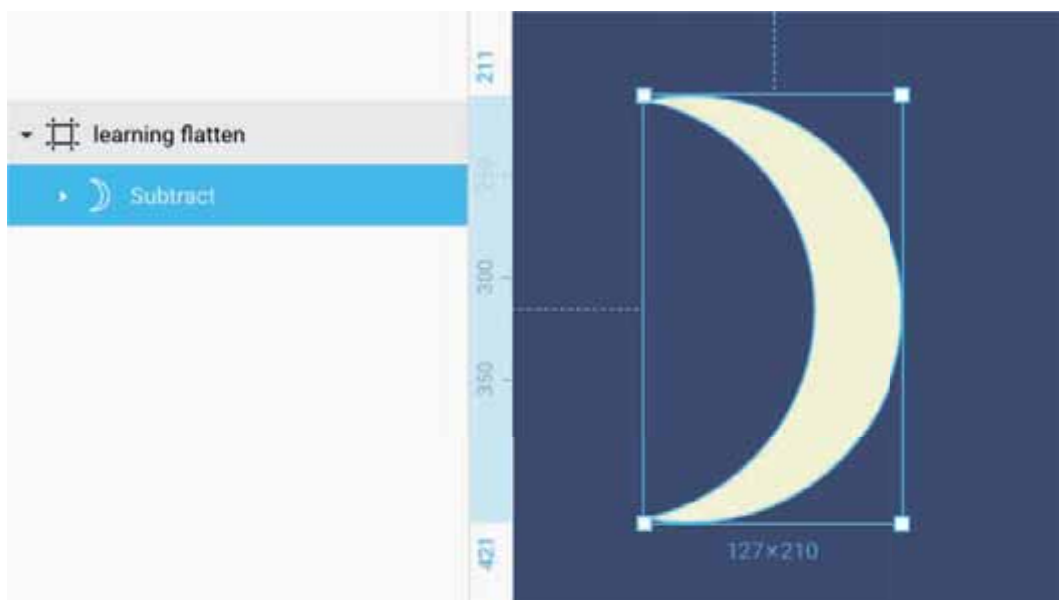
В Скетче эта функция тоже есть и она имеет более понятную клавишу **Ctrl + Opt + F**.

На Маке можно [переназначить](#) на ту же клавишу, которая используется в Скетче. Заметил, что такая клавиша иногда срабатывает со второго раза.

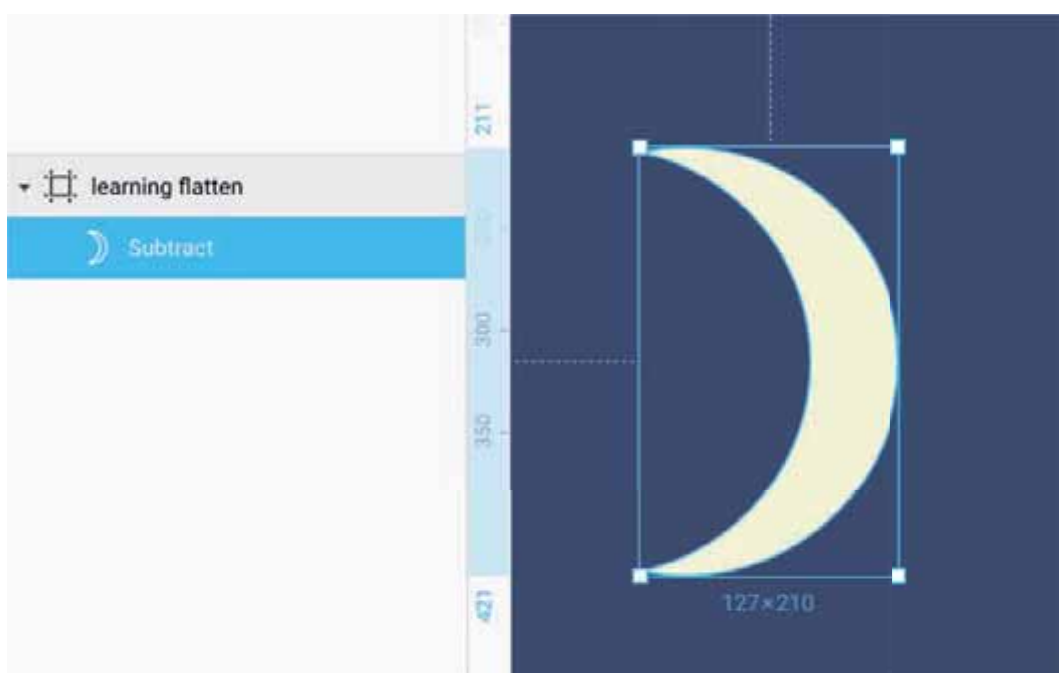
Вызвать **Flatten** можно через поиск: **Cmd + /, fl**.

Как сделать флэтен

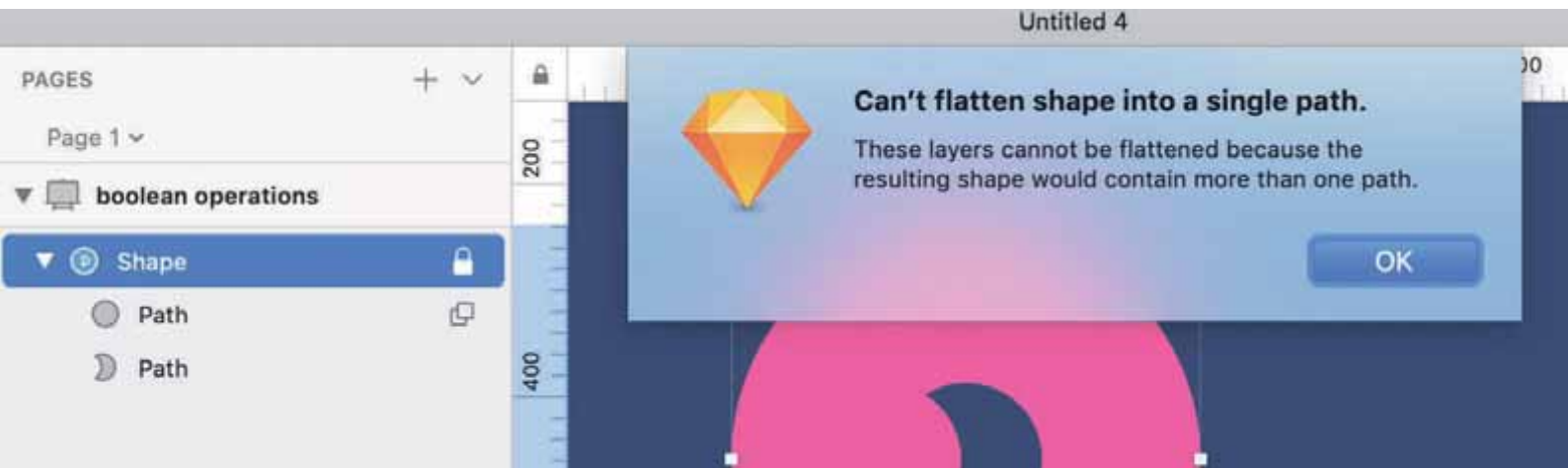
1. Выделяем булеву группу. Обращаем внимание на треугольник слева, который её разворачивает.



2. Нажимаем **Flatten**, которую назначили на **Ctrl + Opt + F**. Раскрывающий треугольник левее названия слоя исчез. Группа превратилась в шейп в форме полумесяца. Теперь его можно менять только в режиме редактирования, **Enter**.



Flatten в Скетче имеет ограничения: нельзя схлопнуть фигуру, которая имеет отверстия:



Причина в том, что Скетч не может обрабатывать шейпы, которые нельзя прорисовать одной замкнутой векторной линией.

В Фигме такого ограничения нет, поскольку плоская фигура может состоять из любого количества несвязанных линий. Благодаря этой же особенности в Фигме реализованы векторные сети.

12. Режим Outline: векторные контуры

Проект в Фигме →

В Фигме есть режим **Outline View**, которого нам многие годы не хватало в Скетче. **Outline** удобно использовать для работы с размерами фреймов и для понимания, из чего состоит векторная форма.

Клавиша та же, что и в Иллюстраторе: **Cmd + Y**.



В этом режиме видны только векторные контуры слоёв без учёта обводок и заливок.

- Фреймы показываются пунктиром. У шейпов видны скругления.
- Наконечники стрелок не показываются.
- У текстовых слоёв видны шейпы букв, но увы, не видны боксы.

Находим невидимые шейпы

Другой частый кейс, в котором удобно использовать режим **Outline** — когда нужно найти невидимую фигуру, у которой нет ни обводки, ни заливки, но осталась векторная форма.

Как выделять труднодоступные слои

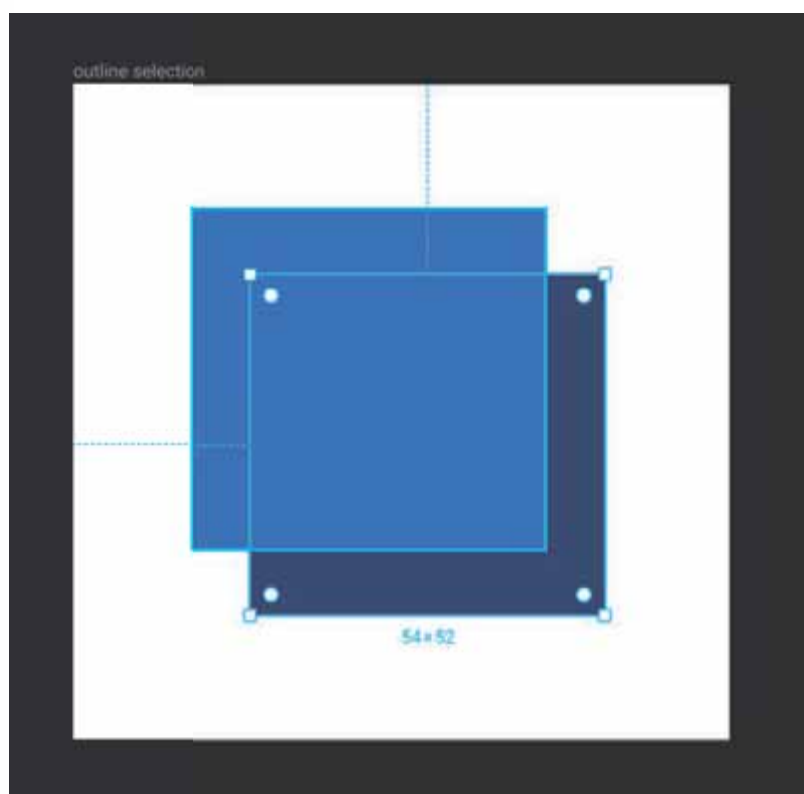
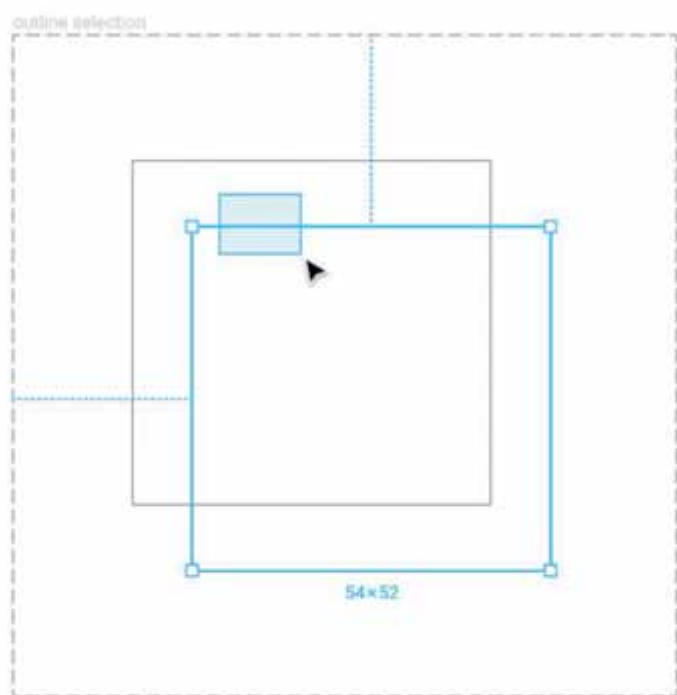
В Иллюстраторе режим **Outline** удобно использовать для выделения труднодоступных слоёв, загороженных другими. Для этого мы кликаем на контур нужного слоя. В Фигме такой фокус не прокатит: верхний слой поймает клик на себя. Но есть решение.

1. Допустим, необходимо выделить нижний квадрат, который загорожен верхним. И сделать это надо сквозь верхний.

outline selection



2. Переключаемся в режим **Outline**, **Cmd + Y**.
3. Зажимаем **Cmd** и растягиваем выделяющий прямоугольник левой кнопкой. В него должен попасть контур нижней фигуры, при этом надо не задеть верхнюю.
4. Выходим из **Outline**, **Cmd + Y**, выделен нижний квадрат.



13. Режимы цветового кодирования

[Проект в Фигме →](#)

Там только иллюстрации.

Эта базовая информация пригодится, когда мы будем изучать заливки, обводки и градиенты. Если ты всё это знаешь, не трать время и листай следующие 5 страниц.

HEX: Шестнадцатичные цвета

Любой цвет, который виден на экране в рамках цветовой системы **RGB** (Red, Green, Blue), может быть передан при помощи комбинации трёх цветовых каналов: красного, зелёного и синего. хекс-коды состоят из трёх пар символов, каждая из которых отвечает за один из этих каналов. Используется шестнадцатичное исчисление: 0 1 2 3 4 5 6 7 8 9 A B C D E F

00 — минимальное значение канала, а FF — максимальное.

#000000 — отсутствие цвета по всем трём каналам, чёрный

#FFFFFF — максимум по всем каналам, белый

#FF0000 — максимум по красному каналу, ярко-красный

#00FF00 — максимум по зелёному, ярко-зелёный

#0000FF — максимум по синему каналу

Если по всем трём каналам стоит равное значение, например, **#808080**, цвет будет монохромным, то есть не будет иметь оттенка. Это происходит, потому что каналы друг друга компенсируют.

Хекс-коды — это основной стандарт кодирования цветов в вебе. Их плюс в том, что их удобно копировать и передавать.

RGB-цвета

Помимо хекс-кодов, цвета можно кодировать в похожей трёхканальной системе, где вместо кодов используются десятичные значения.

0 — минимальное значение канала, 255 — максимальное.

0, 0, 0 — отсутствие цвета по всем каналам, чёрный

255, 255, 255 — максимум по всем каналам, белый

255, 0, 0 — максимум по красному каналу, ярко-красный

0, 255, 0 — максимум по зелёному, ярко-зелёный

0, 0, 255 — максимум по синему каналу

Такой метод кодирования цветов применяется в печати наравне с системой каналов **CMYK** (Cyan, Magenta, Yellow, Black), а также послужил прототипом для системы RGBa, о которой поговорим дальше.

Плюс такой системы, что такие коды более запоминаемы и их удобно диктовать на слух.

RGBa-коды

В качестве альтернативы неинтуитивным хекс-кодам разработали систему кодирования, адаптированную для использования в коде веб-страниц и иных программ. Она содержит в основе RGB, а также добавляет к этим трём каналам дополнительный канал, который позволяет задать опасность. Он называется альфа-каналом.

Первые три канала задаются от 0 до 255, а альфа-канал от 0 до 1.

Значение **0.5** соответствует 50% опасности.

Как формируется RGBa-код:

`rgba(красный, зелёный, синий, альфа)`

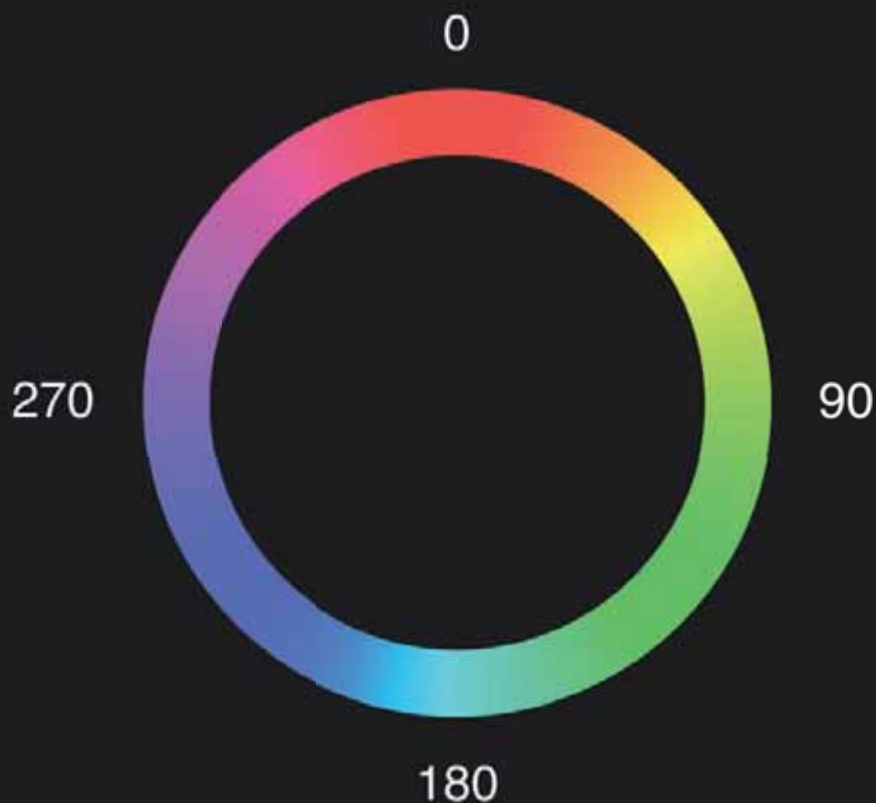
Примеры:

`rgba(0, 0, 0, 0.5)` — чёрный, опасный на 50%

`rgba(255, 255, 255, 1)` — непрозрачный белый

Такая система имеет плюсы от обоих подходов: RGBa-код можно скопировать и всё ещё легче запоминать.

Их можно вставлять в CSS-код вместо хексов.



HSB-цвета

Существует также третья система кодирования цвета – HSB.

В ней цвет не делится на каналы, а формируется из трёх качеств:

Hue [хью], Saturation [сэтьюрэйшн], Brightness [брайтнес].

1. **Hue** — оттенок по цветовому кругу, который соответствует градусу. От 0° до 359° . Гораздо нагляднее эта шкала будет, если представить её в виде круга. Шкала начинается с красного. Бирюзовому оттенку, противоположному красному соответствует угол 180. Увы, это неочевидно из шкалы оттенка:



2. **Saturation** — насыщенность оттенка от 0 до 100.

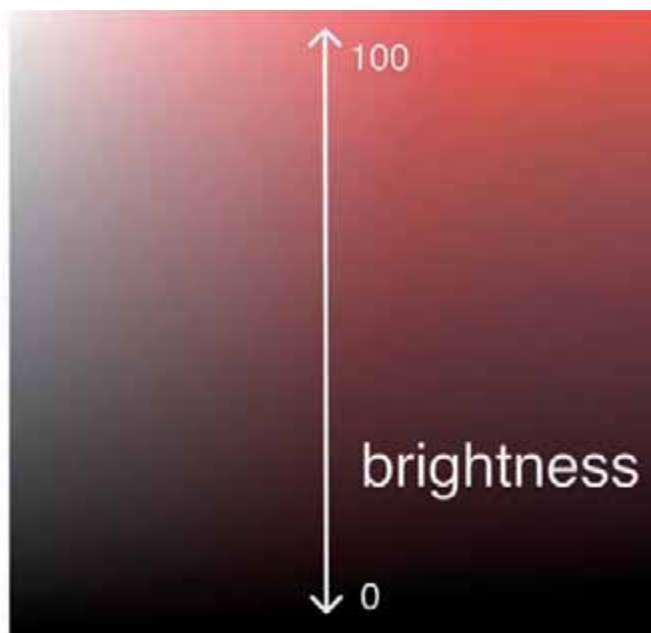
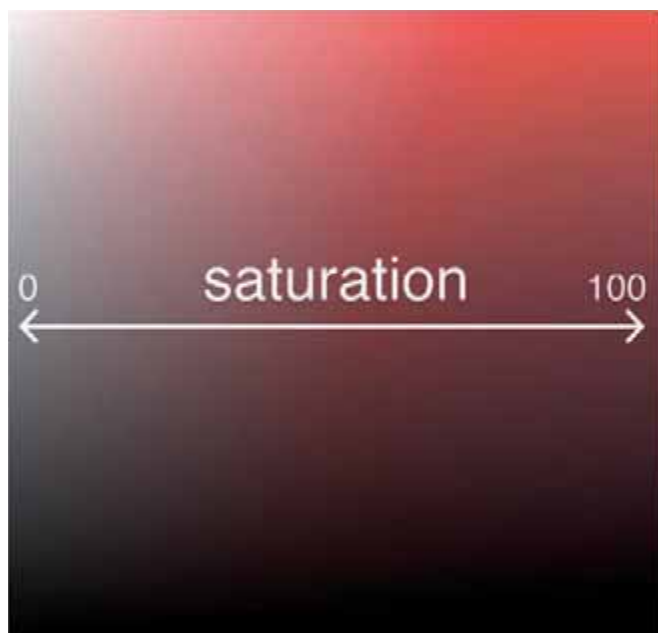
Передаёт, насколько цвет *едкий* и интенсивный.

Цвет может быть бледным или концентрированным. Если у него вовсе отсутствует оттенок, значит, он монохромный и в шкале цветов будет прибит к левой границе. Чем больше насыщенность, тем ближе к правой. Чтобы менять насыщенность, нужно передвигать маячок по горизонтали.

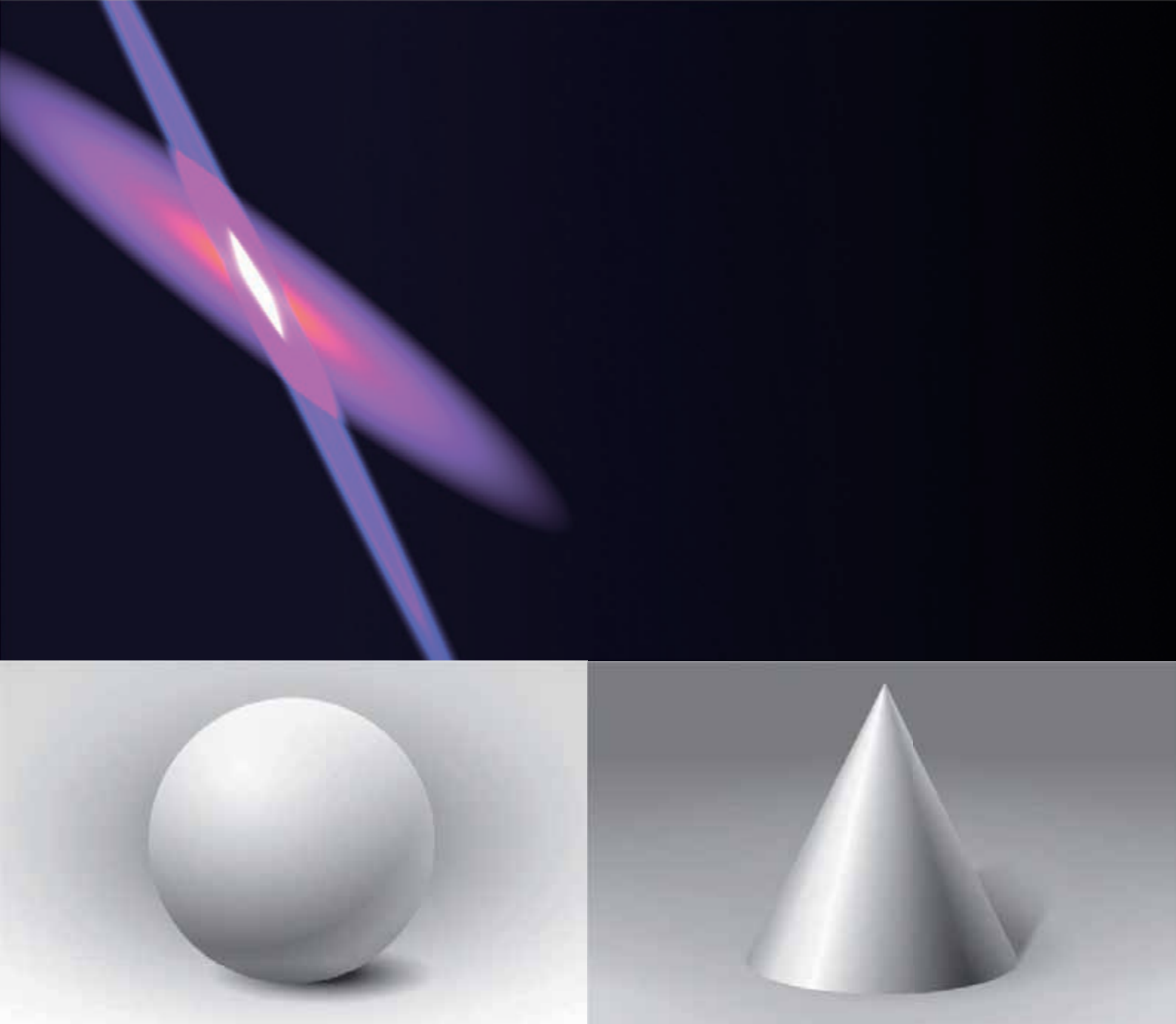
3. **Brightness** — яркость оттенка. Какова доля чёрного в оттенке.

Красный может стремиться к бордовому, и это значит, что в нём подмешано много чёрного, следовательно, понижена яркость. Напротив, чистый красный показывает отсутствие чёрного. Чтобы менять яркость, нужно передвигать маячок вертикально.

Шкала красного оттенка:



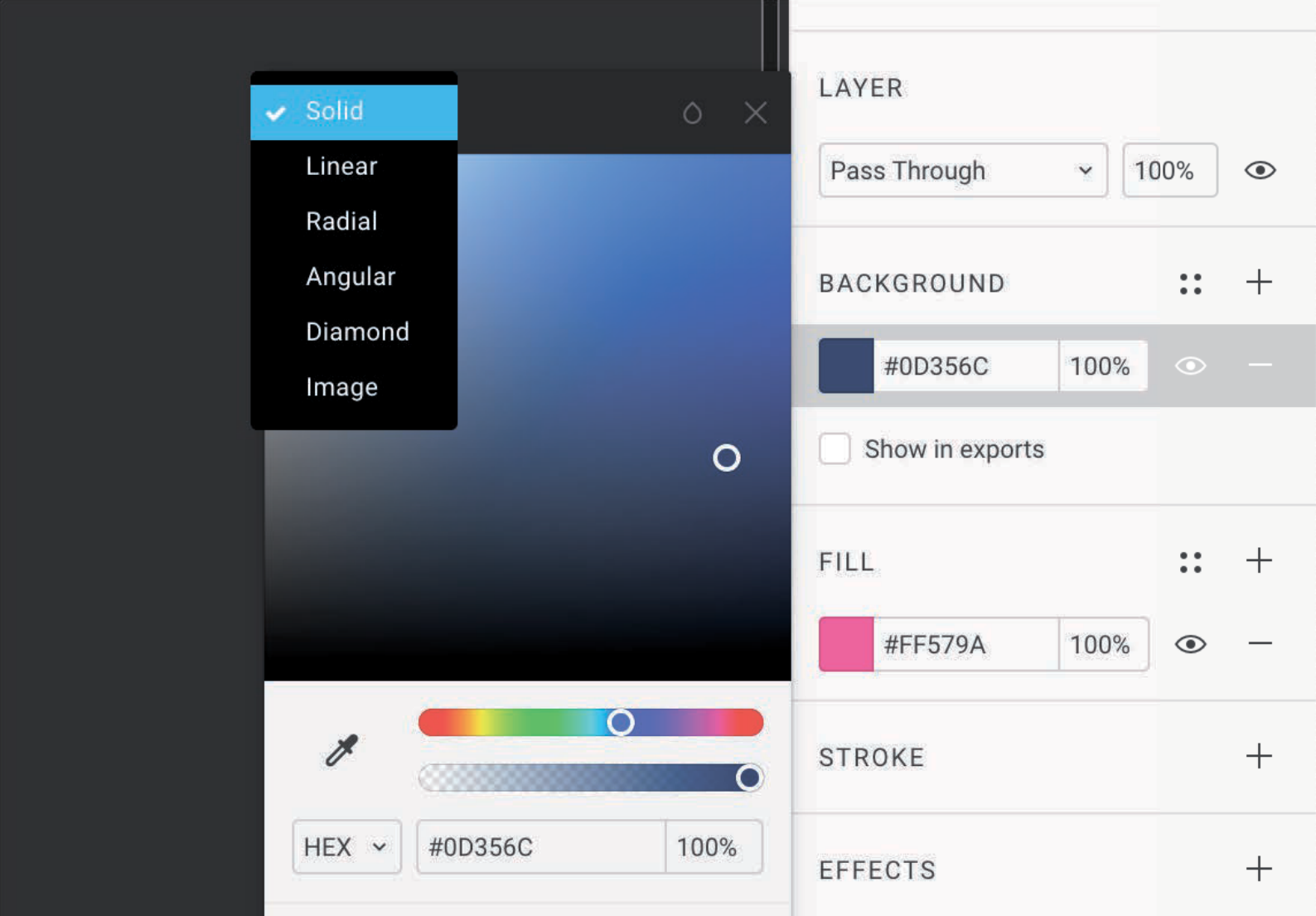
На этом с теорией всё.



14. Заливка и градиенты

[Проект в Фигме →](#)

Заливка **Fill** может быть у фрейма, шейпа или текстового слоя. В этой главе рассмотрим, какие бывают режимы заливки и научимся их применять на практике.



В верхнем левом углу развёрнутой палитры отображается режим заливки фона:

Левее радужного фейдера размещена кнопка с пипеткой, позволяющей снимать цвета в любом месте холста. У пипетки клавиша как в Скетче:

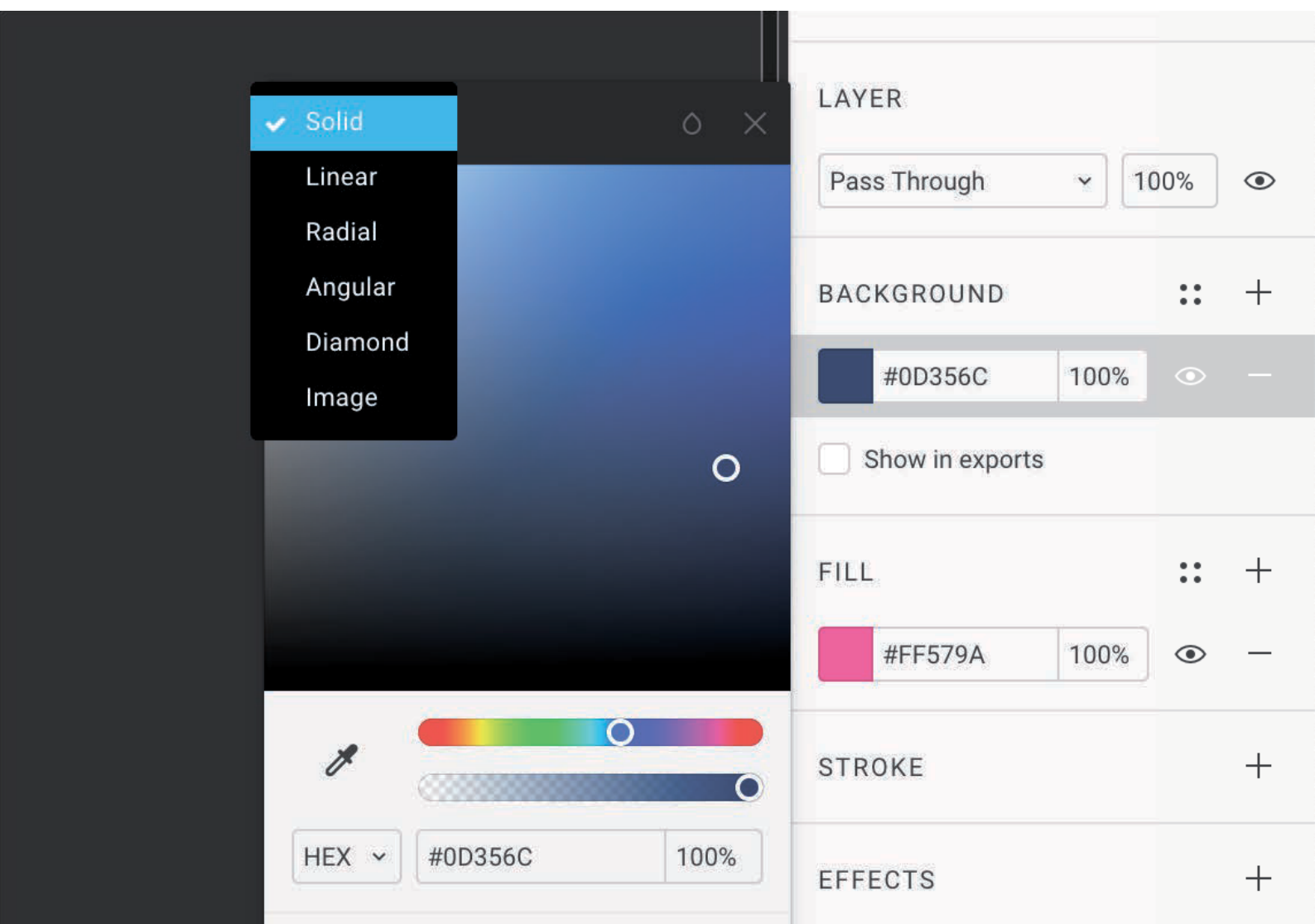
Ctrl + C. Мнемоника: **Color**. На Windows – **I**.

За исключением **Diamond**, режимы заливки в Фигме и Скетче очень похожи:

- **Solid** [сóлид] — режим заливки сплошным цветом, который можно задавать как шестнадцатичный код (HEX-код от слова *hexadecimal*) от белого **#FFFFFF**, до **#000000** чёрного.
- **Linear** [линиар] — режим заливки линейным градиентом, в котором могут быть две или больше контрольных точек цвета.
- **Radial** [рэдиал] — режим заливки радиальным градиентом, в котором один конец шкалы градиента ассоциируется с центром круга, а другой с его краями. От центра к краям можно настраивать плавный переход.
- **Angular** [энгюлар] — режим заливки градиентом, в котором шкала от начала до конца градиента вращается вокруг определённой точки на заливаемой фигуре.
- **Diamond** [даймонд] — режим заливки радиальным градиентом с 4 лучами, который есть только в Фигме. Позволяет настраивать градиенты с прямоугольной текстурой.
- **Image** — режим заливки растровым изображением. Применяется для вставки растровых слоёв в Фигму. Если вставить PNG или JPG-файл в окно редактора, он появится на холсте как прямоугольник с заливкой этого типа.

Режим заливки Solid: ровный цвет

Простейший режим заливки сплошным цветом. Изначально цвет показан в виде хекс-кода. Такие коды не несут информации о том, что цвет может быть прозрачным.

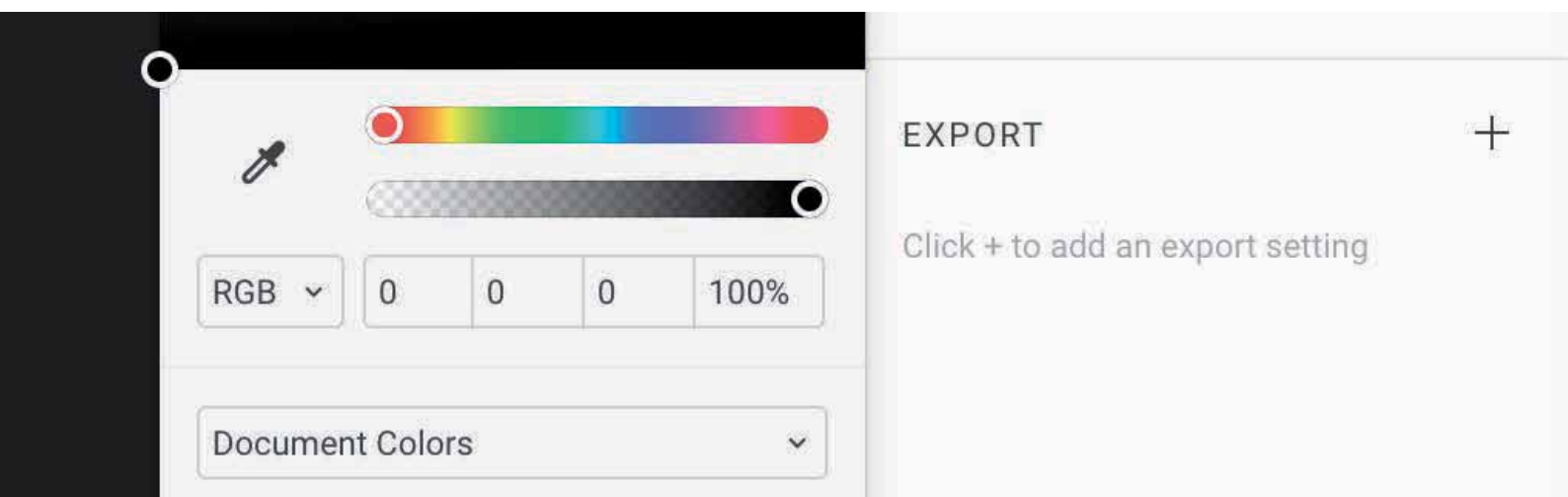
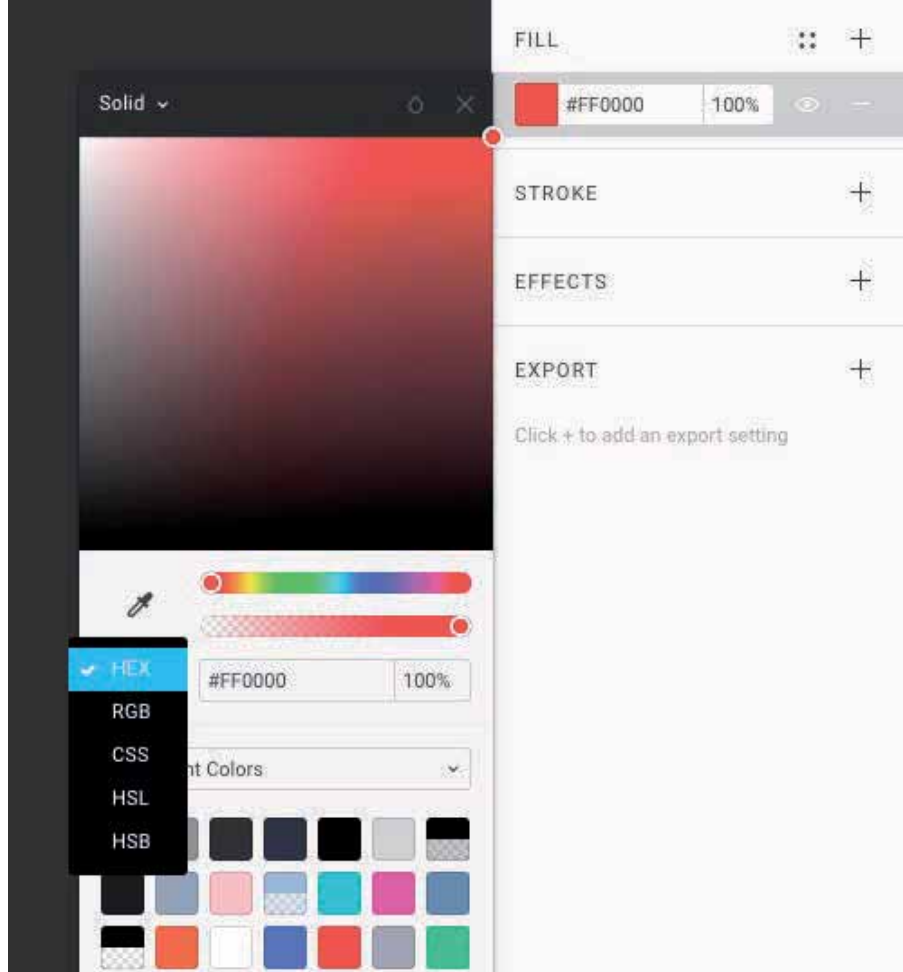


Фейдер ниже радужного позволяет настроить прозрачность заливки от 0 до 100. Также его можно задать в поле справа от хекс-кода.

Прозрачность слоя настраивается в похожем поле в блоке **Layer** напротив режима наложения. Сейчас установлен режим по умолчанию **Pass Through** [пэс тру]. Клавиши: **1 - 0**.

В режиме **Solid** можно переключить способ кодирования цвета:

В режиме RGB мы видим три поля для каждого канала и четвёртое поле для альфа-канала.

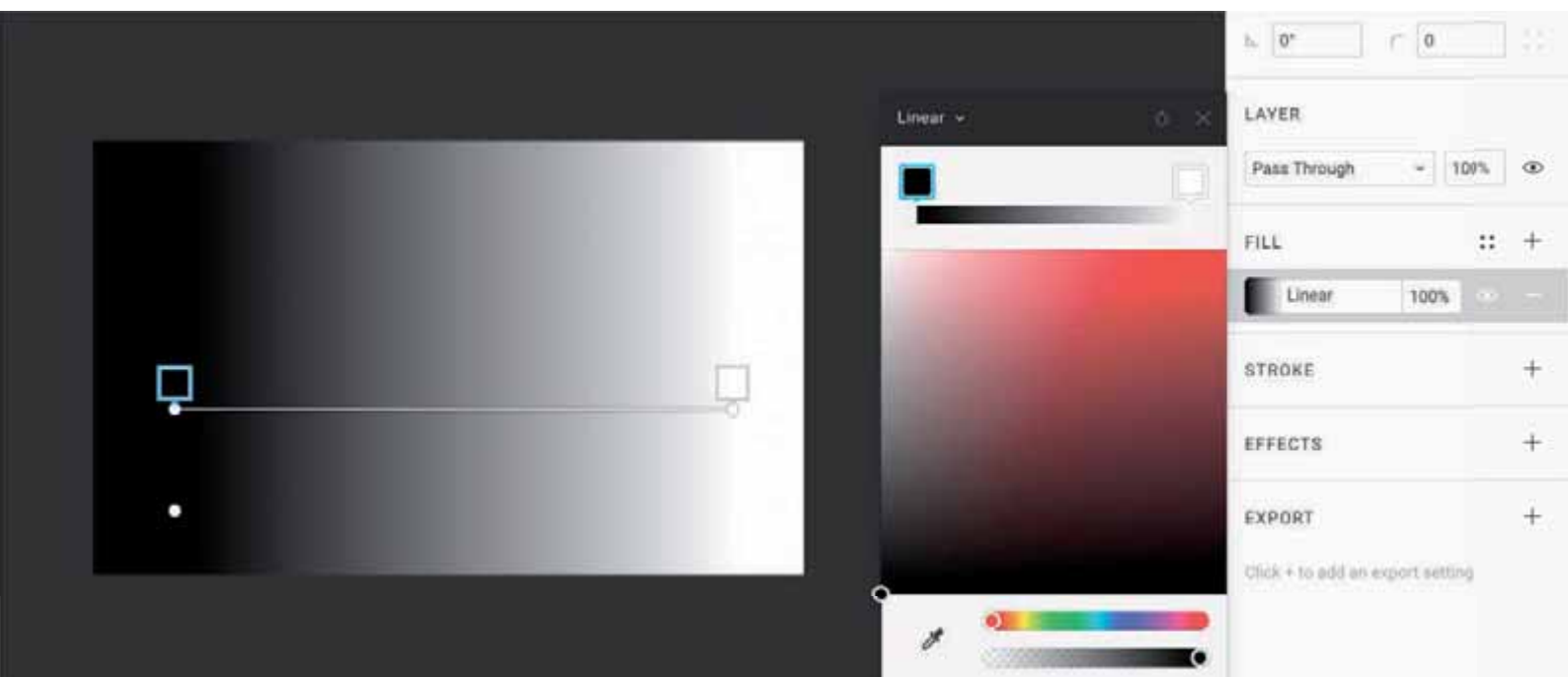


Режим заливки Linear: линейный градиент

Градиенты — плавные переходы от одного цвета к другому.

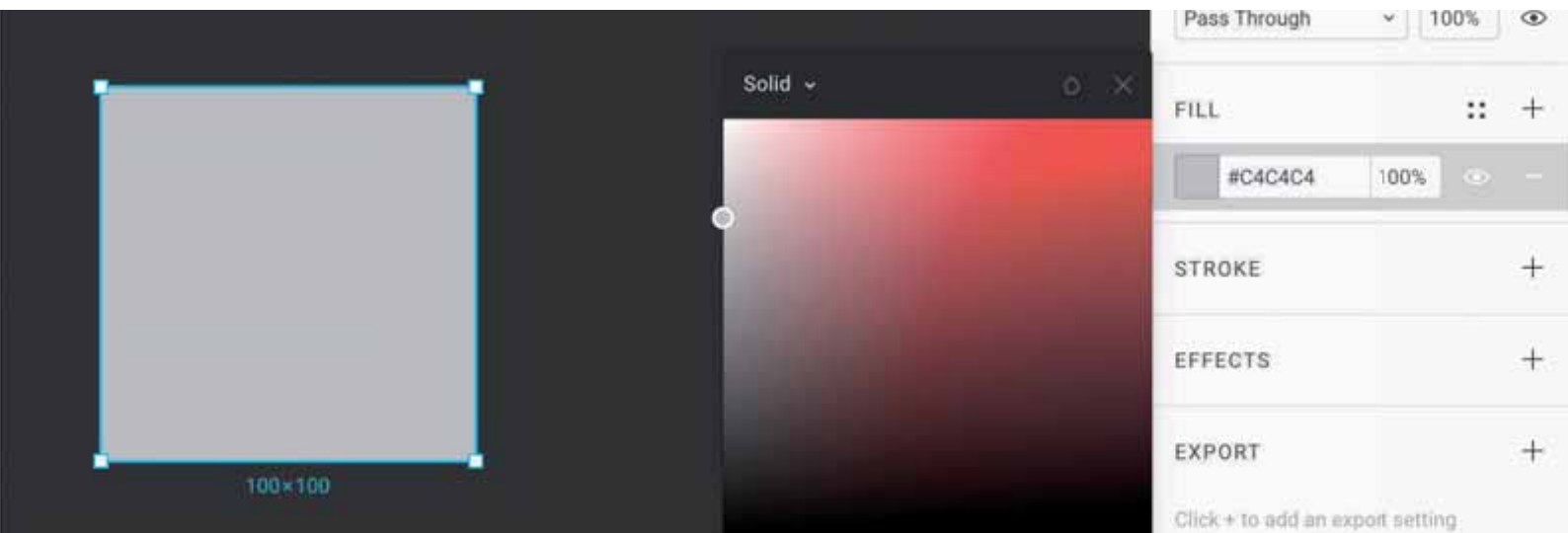
Линейный градиент имеет ось, вдоль которой меняется цвет.

Задать, какие цвета участвуют в градиенте, можно при помощи контрольных точек на его оси. В этом простейшем градиенте их две: чёрная и белая.

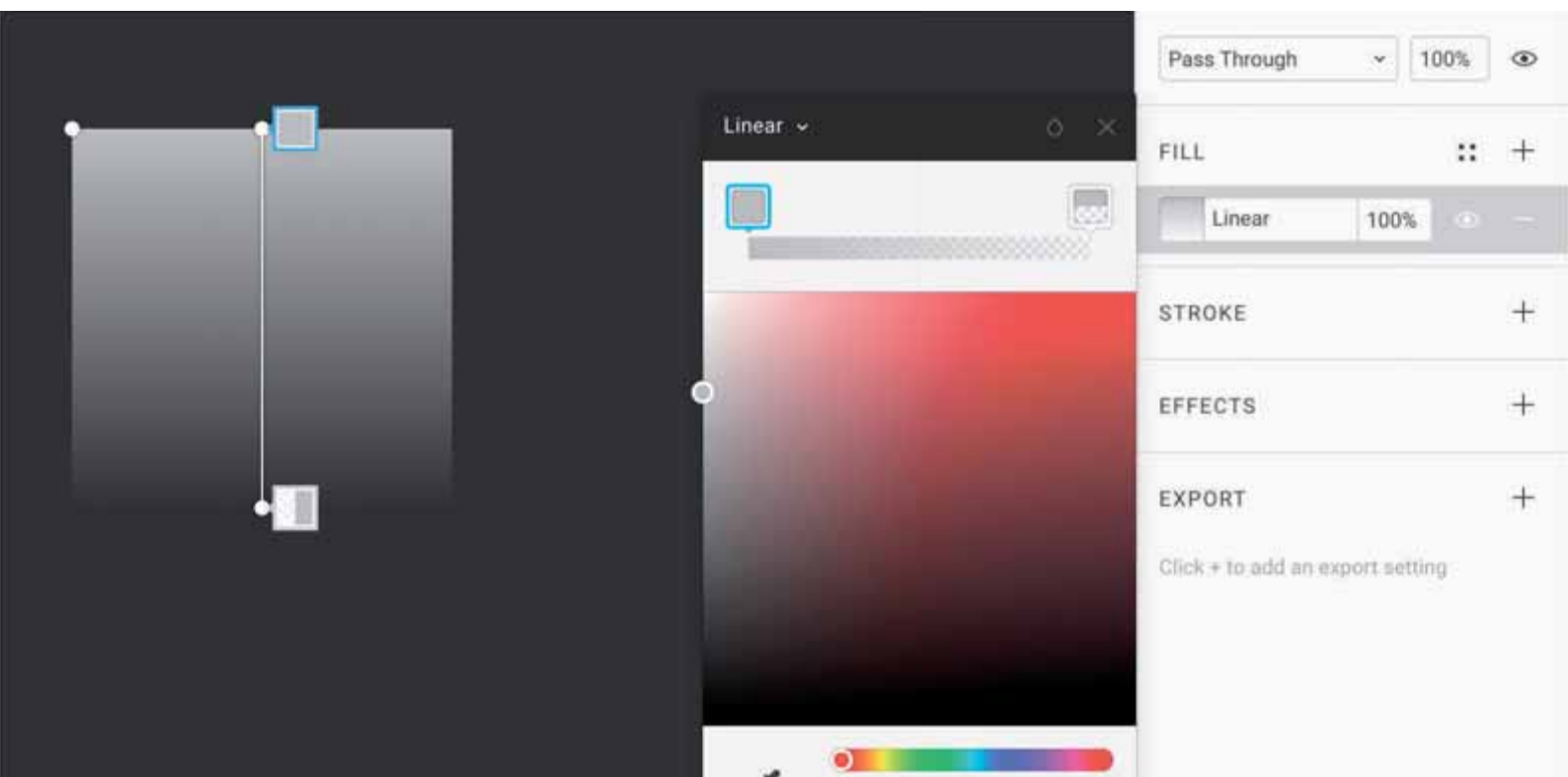


Создадим градиент:

1. Создадим шейп квадрата, **R**. Развернём окно заливки.
По умолчанию её цвет серый #C4C4C4.
2. В левом верхнем углу меняем режим заливки с **Solid** на **Linear**.



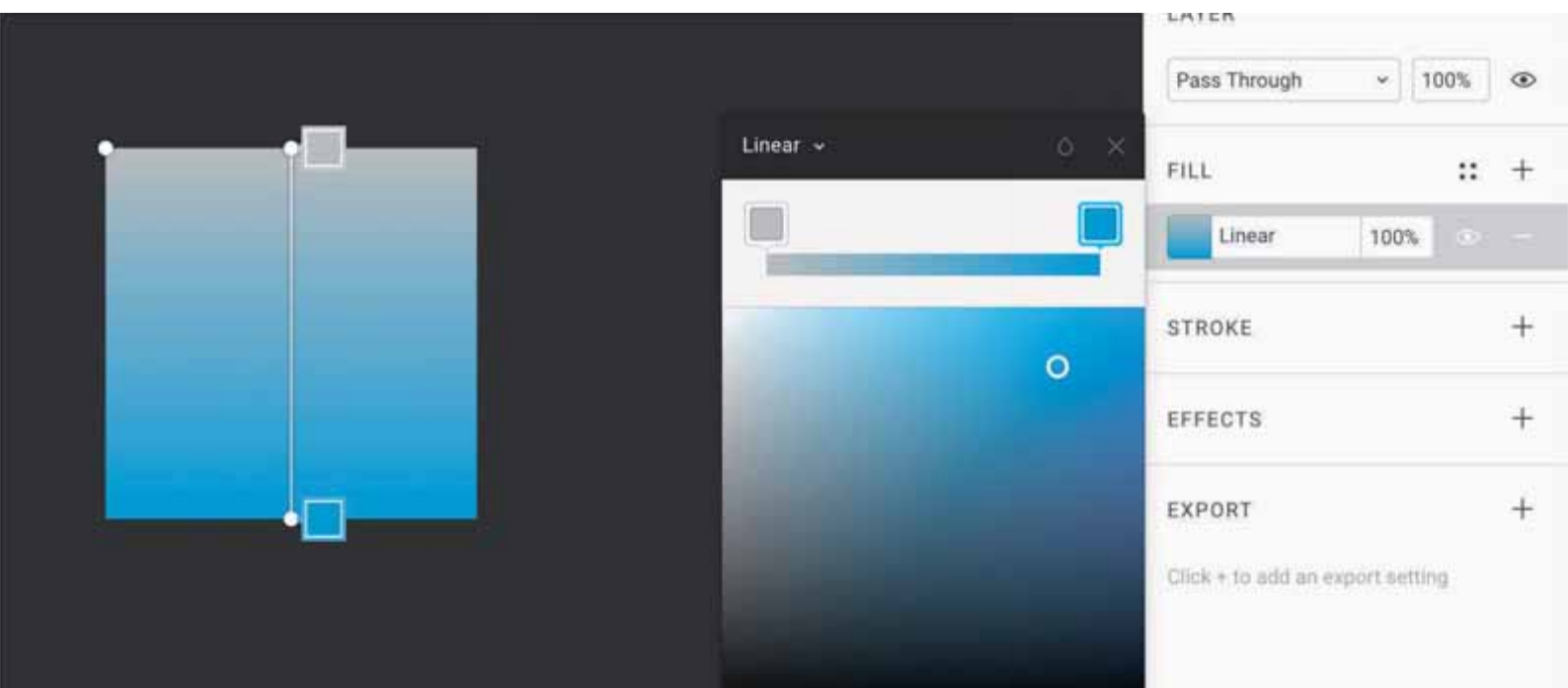
По центру фигуры появилась вертикальная ось и две контрольные точки цветов. Верхняя точка выделена.



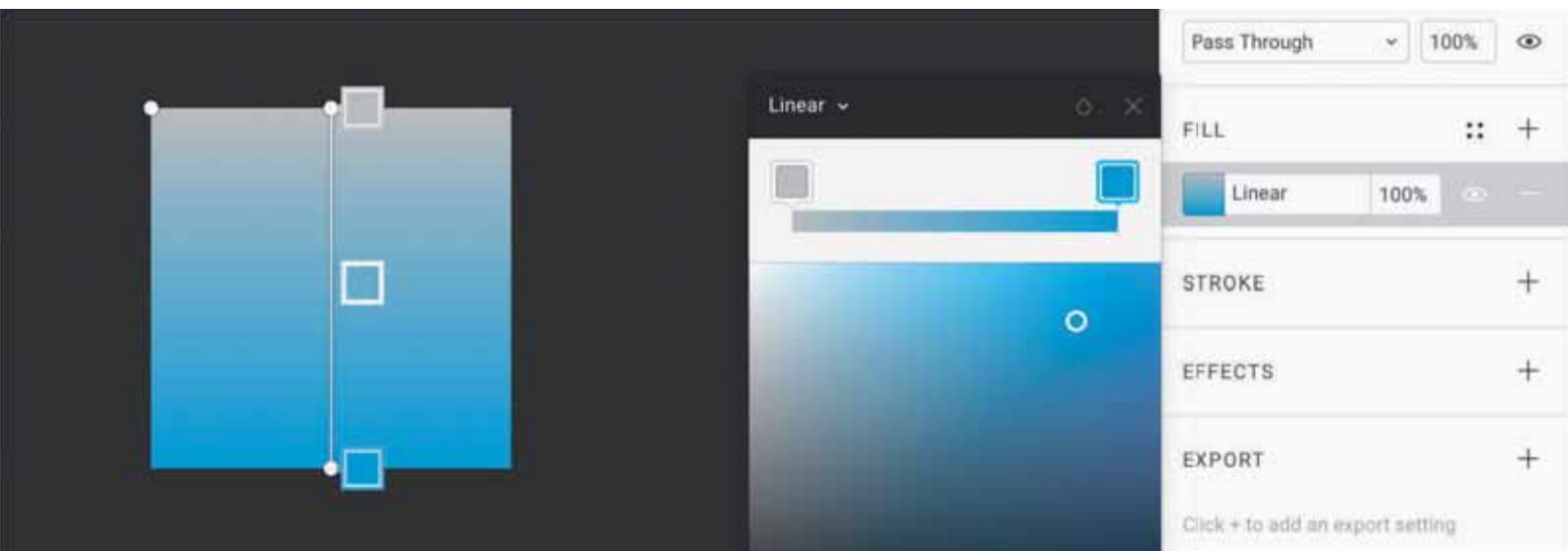
В нижней точке выбран тот же цвет, однако его опасность выкручена в 0. Поэтому, возникает эффект плавного исчезновения квадрата. Нижний квадратный пробник цвета разделён на две части. В левой половине виден цвет с учётом текущей настройки опасности, справа полностью опасный.

Если в режиме **Solid** был изначально выбран другой цвет, при переходе в **Linear** он будет использован в градиенте.

3. Выделим нижнюю точку, вернём опасность и поменяем цвет. Станет явно виден переход между цветами. В момент подбора оттенка рекомендую покрутить радужный слайдер оттенка (hue). Это помогает подбирать такие оттенки, которые не конфликтуют друг с другом. Если указатель находится над палитрой, оттенок можно настраивать скроллом мыши.



4. В центре оси можно кликнуть и таким образом добавить ещё одну контрольную точку цвета. Если нужно её удалить, нажимаем **Delete**.



Точки можно передвигать по оси, а саму её тянуть за концы и задавать ей нужный угол.

Полезные клавиши для работы с градиентами

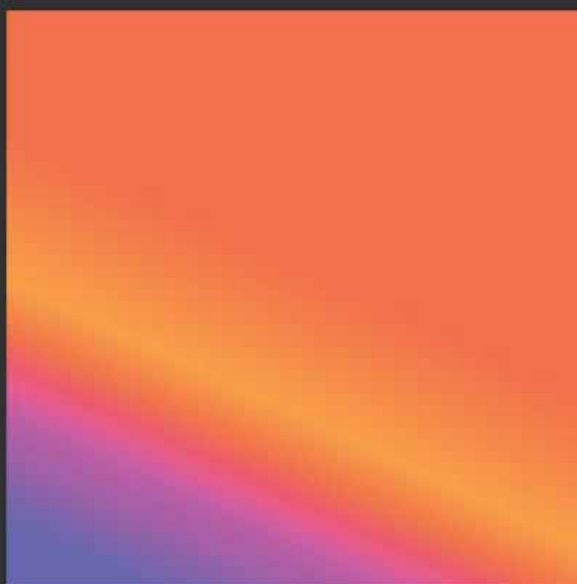
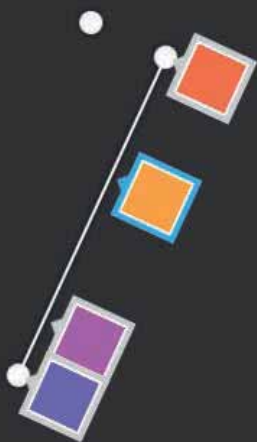
Чтобы зафиксировать градиент и скрыть панель заливки, нажимаем **Enter** или **Esc**.

Если контрольная точка выделена, её можно смещать **стрелками** по оси градиента. Зажатый **Shift** ускоряет движение стрелки на оси.

Увы, в отличие от Скетча, между точками в градиенте нельзя переключаться с клавиатуры через **Tab** и **Shift + Tab**.

Во время изменения угла зажатый **Shift** позволяет перейти в режим **Snap To Geometry** и фиксировать ось под ровными углами с шагом в 45° ровно так же, как мы это делали в главе про перо.

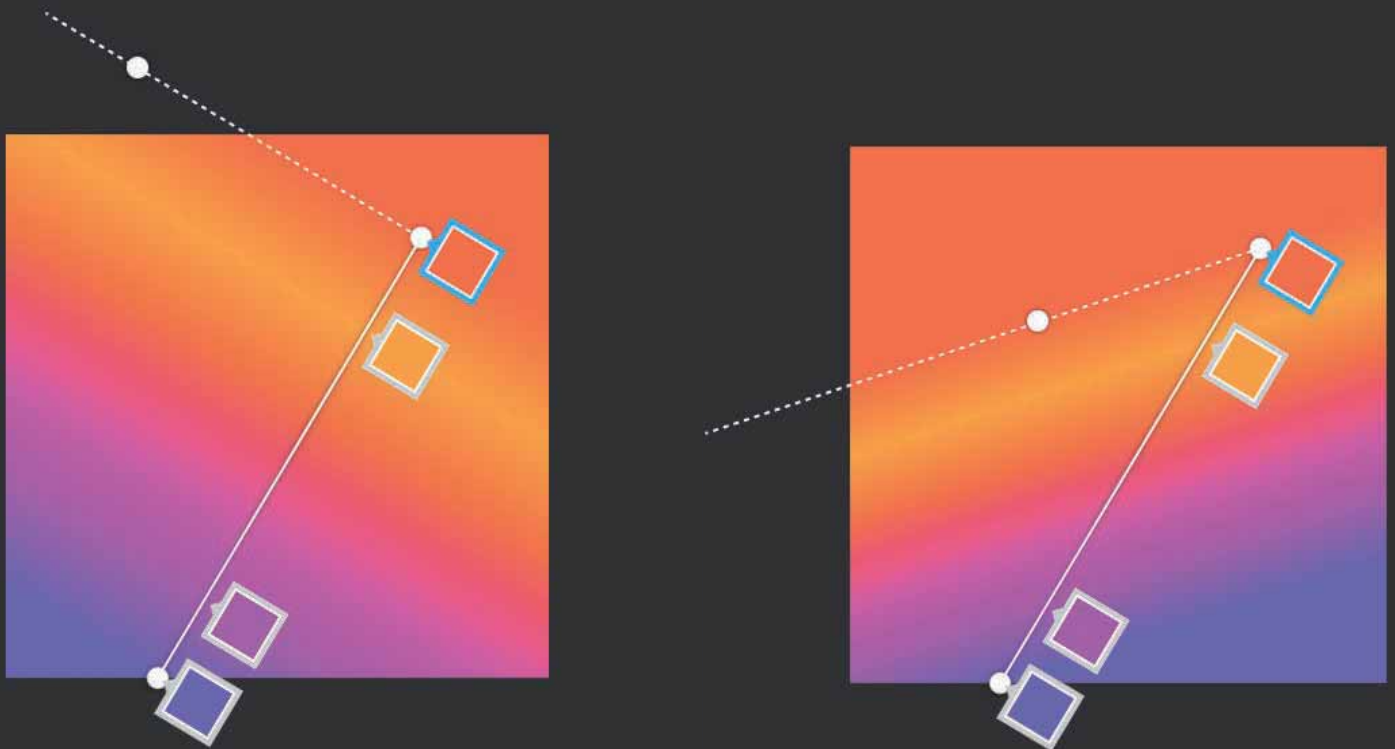
Интересная возможность, которой нет в Скетче: если схватить за один конец линии и зажать **пробел**, всю линию вместе с её контрольными точками можно будет сдвигать, не меняя угла. Это даёт возможность быстро и точно настраивать положение градиента, не двигая каждый его конец индивидуально.



Перпендикуляр градиента

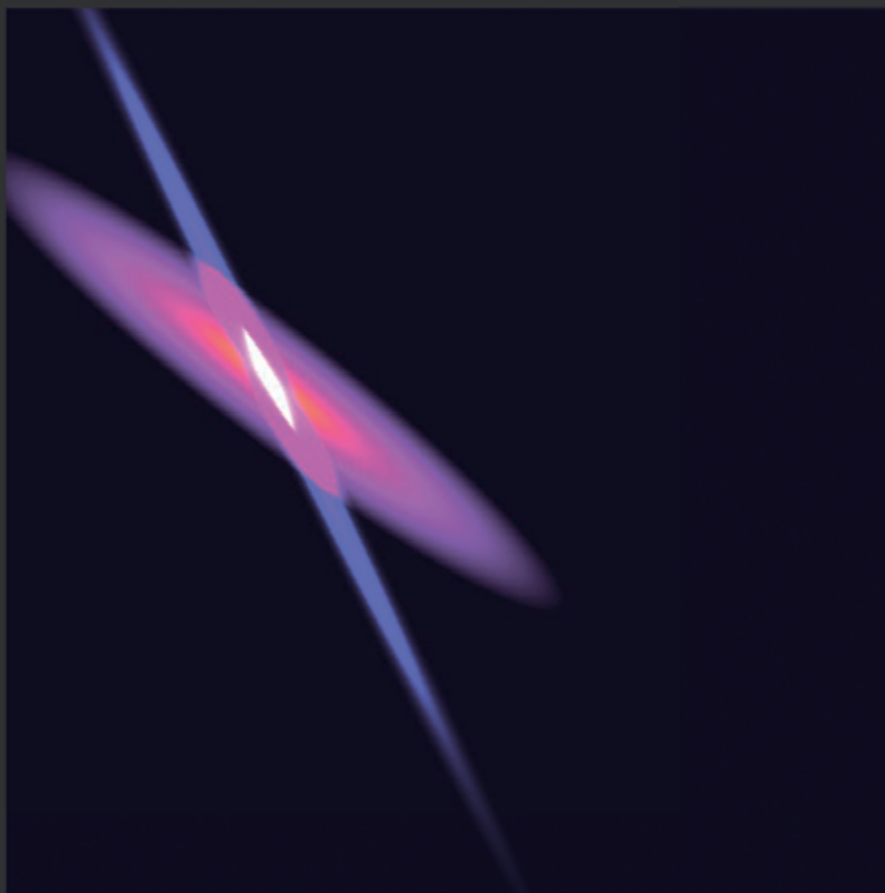
Ещё одна уникальная особенность линейных градиентов в Фигме: угол градиента можно поворачивать относительно основной оси.

У градиента есть третья неприметная круглая ручка, назначение которой сразу непонятно. Если провести через неё прямую линию до одного из концов оси, получится угол 90° . Это дополнительная ручка перпендикулярной оси градиента. Если зажать **Opt** и потянуть за неё, цветовой переход будет отрисован перпендикулярно ей.



Режим заливки Radial: радиальный градиент

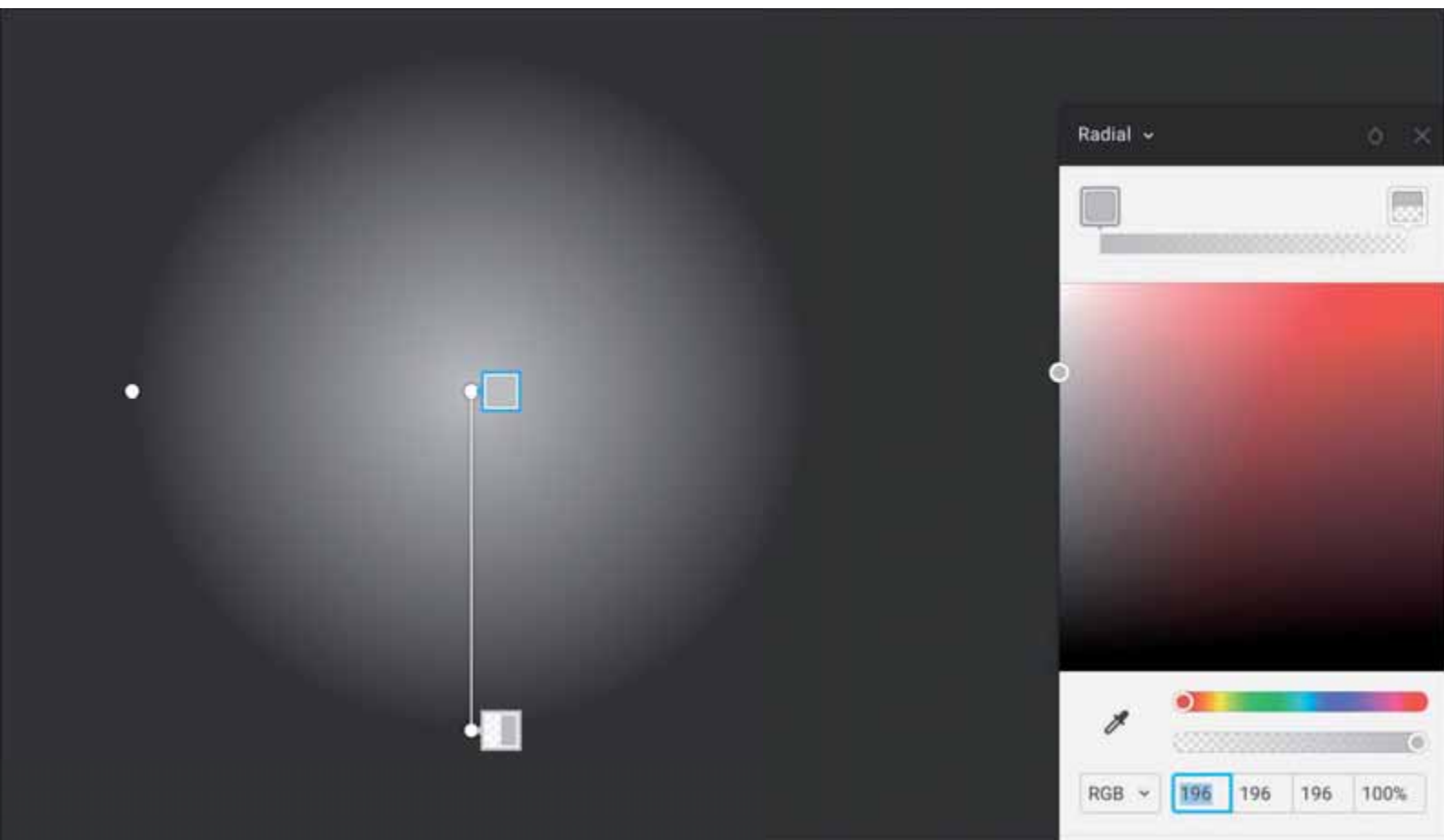
Такой градиент формируется на основе эллипса. У него есть центр и внешняя граница. Они соединены радиусом, на котором размещаются контрольные точки цвета.



Пример из двух наложенных друг на друга радиальных градиентов. Для усиления эффекта свечения применил режим наложения Color Dodge у верхнего.

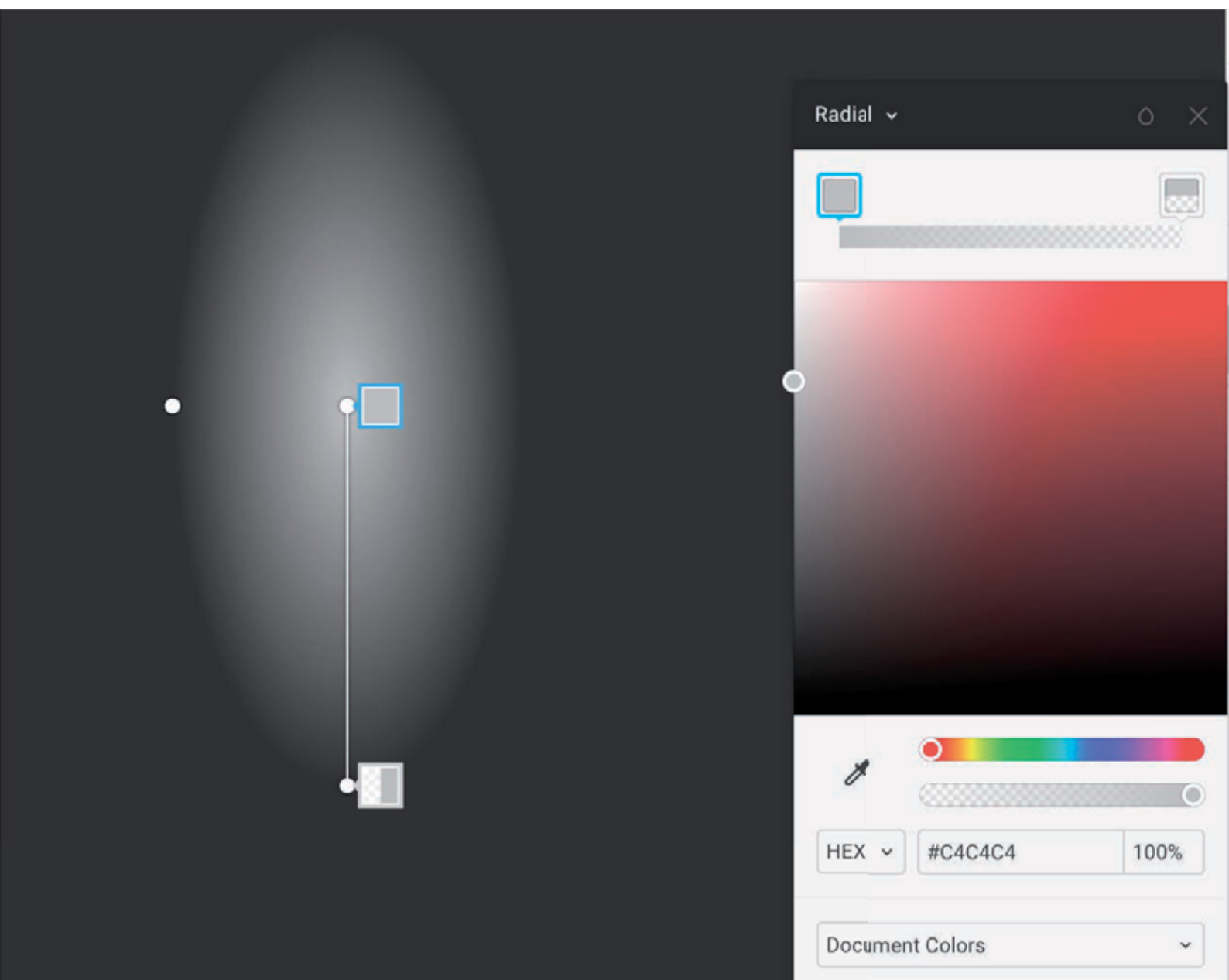
Создадим простейший радиальный градиент:

1. Создадим новый квадрат, **R**. Развернём панель **Fill**.
2. Переведём заливку из режима **Solid** в **Radial**. Результат:



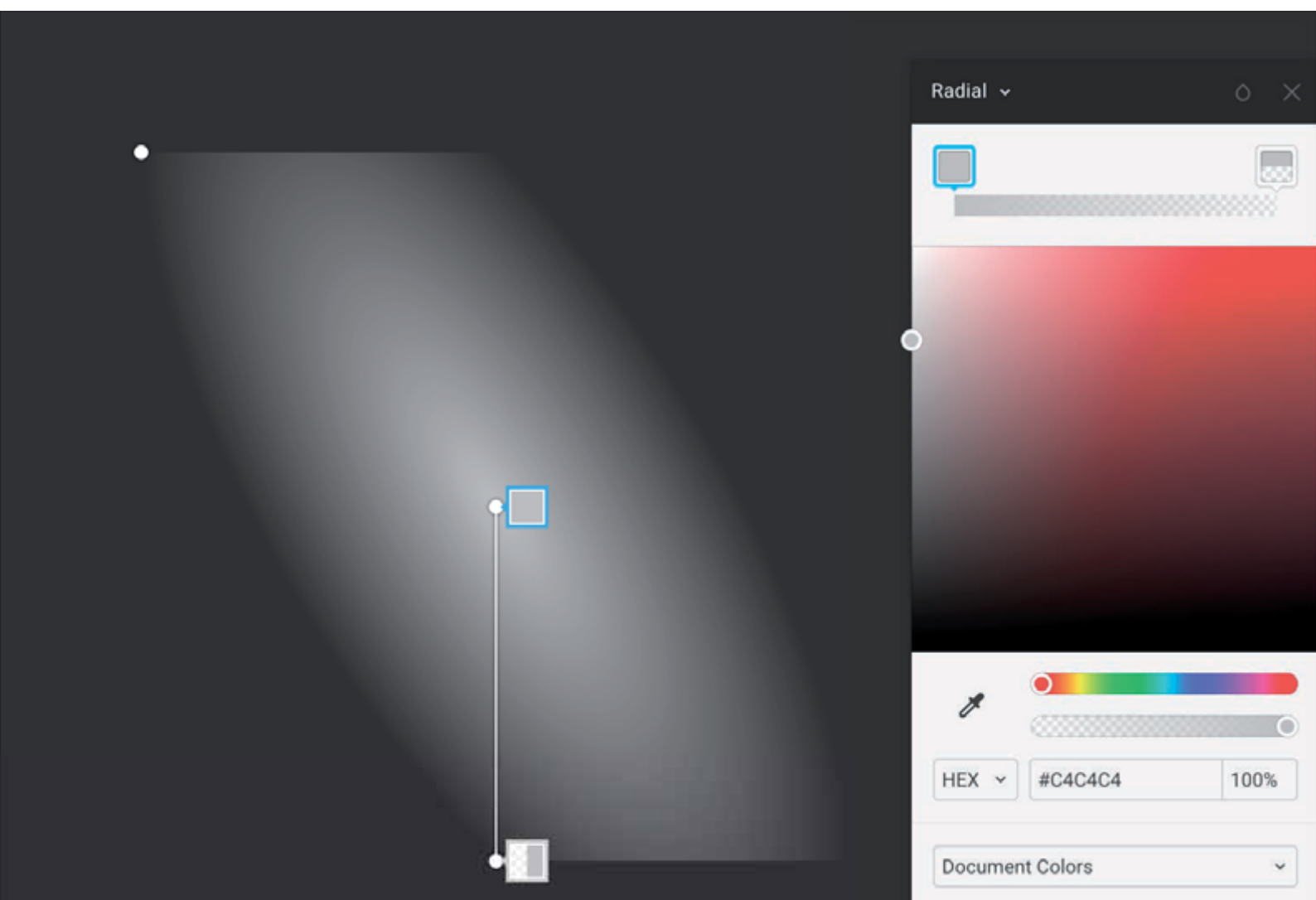
3. Аналогично линейным градиентам у нас есть две контрольные точки серого цвета, одна из которых полностью прозрачна. Также видна радиальная ось.

4. Чтобы создавать радиальные градиенты не только на основе круга, потянем ручку слева. Чтобы сплюснуть круг в эллипс, тянем к центру.

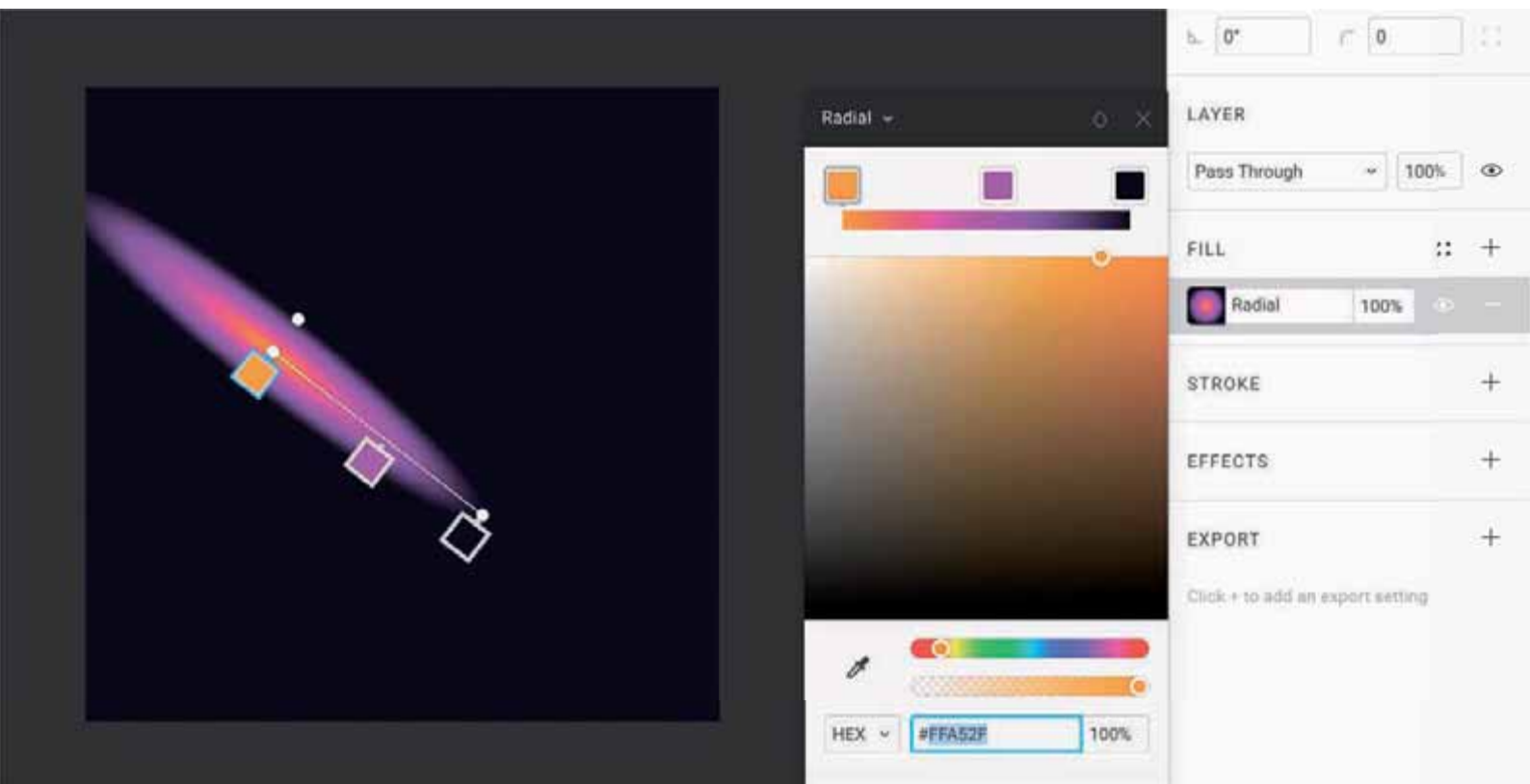


5. Работает также и трюк с **Opt**, как в линейных градиентах. Он позволяет смещать ручку так, чтобы эллипс искажался. Это удобнее, чем настраивать общий угол наклона радиуса, а затем ужимать эллипс по малому радиусу.

В данном примере я с зажатым **Opt** увёл ручку в верхний угол. Эллипс оказался наклонён, хотя основная ось осталась вертикальной.

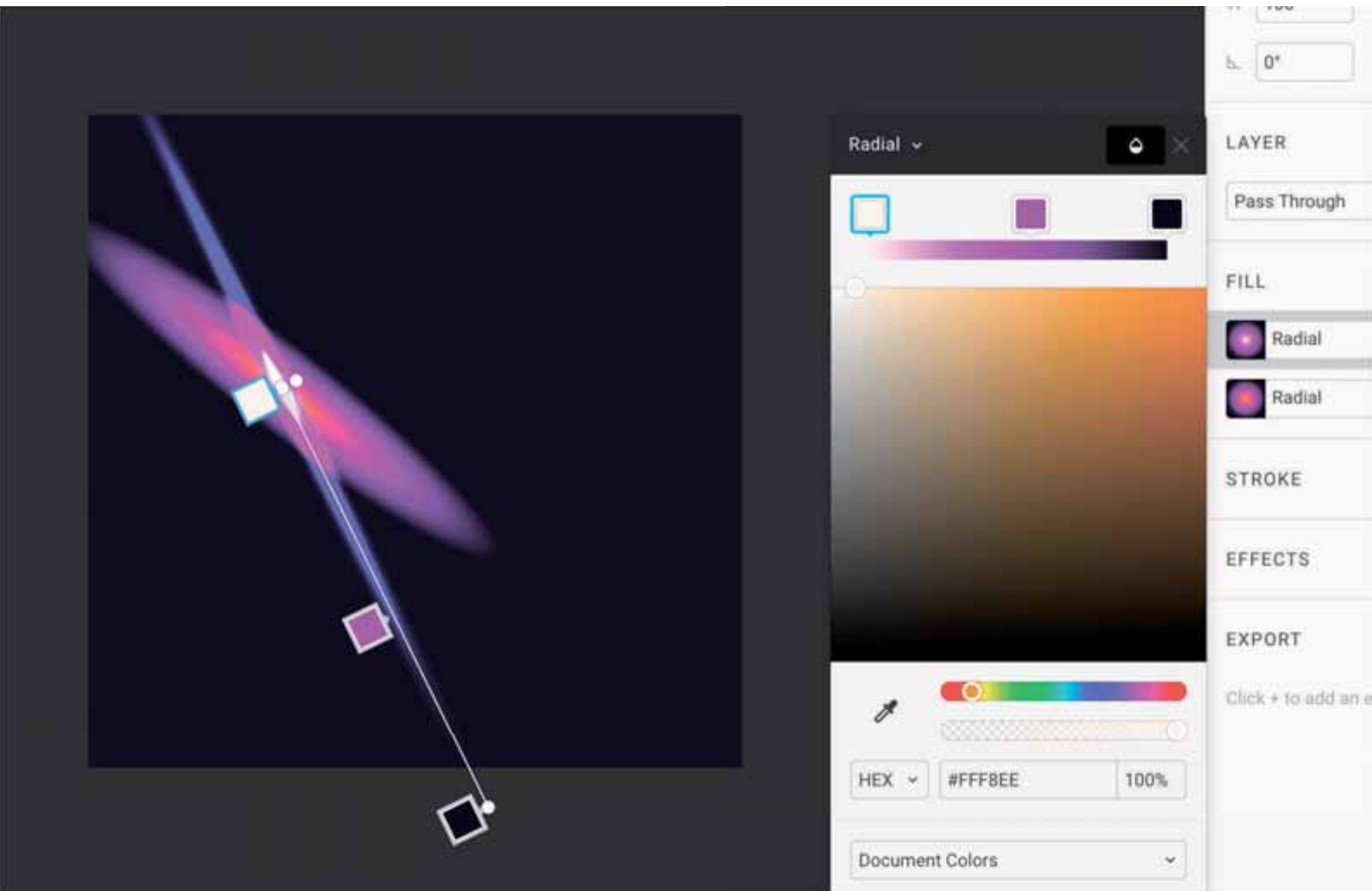


Повторим пример с фиолетовым бликом



1. Накладываем нижний радиальный градиент. Чтобы смещать всю ось, не меняя угла, зажимаем **Shift**.
2. Делаем радиус меньше размера основной фигуры, так чтобы внешний цвет градиента стал фоновым для всей фигуры.

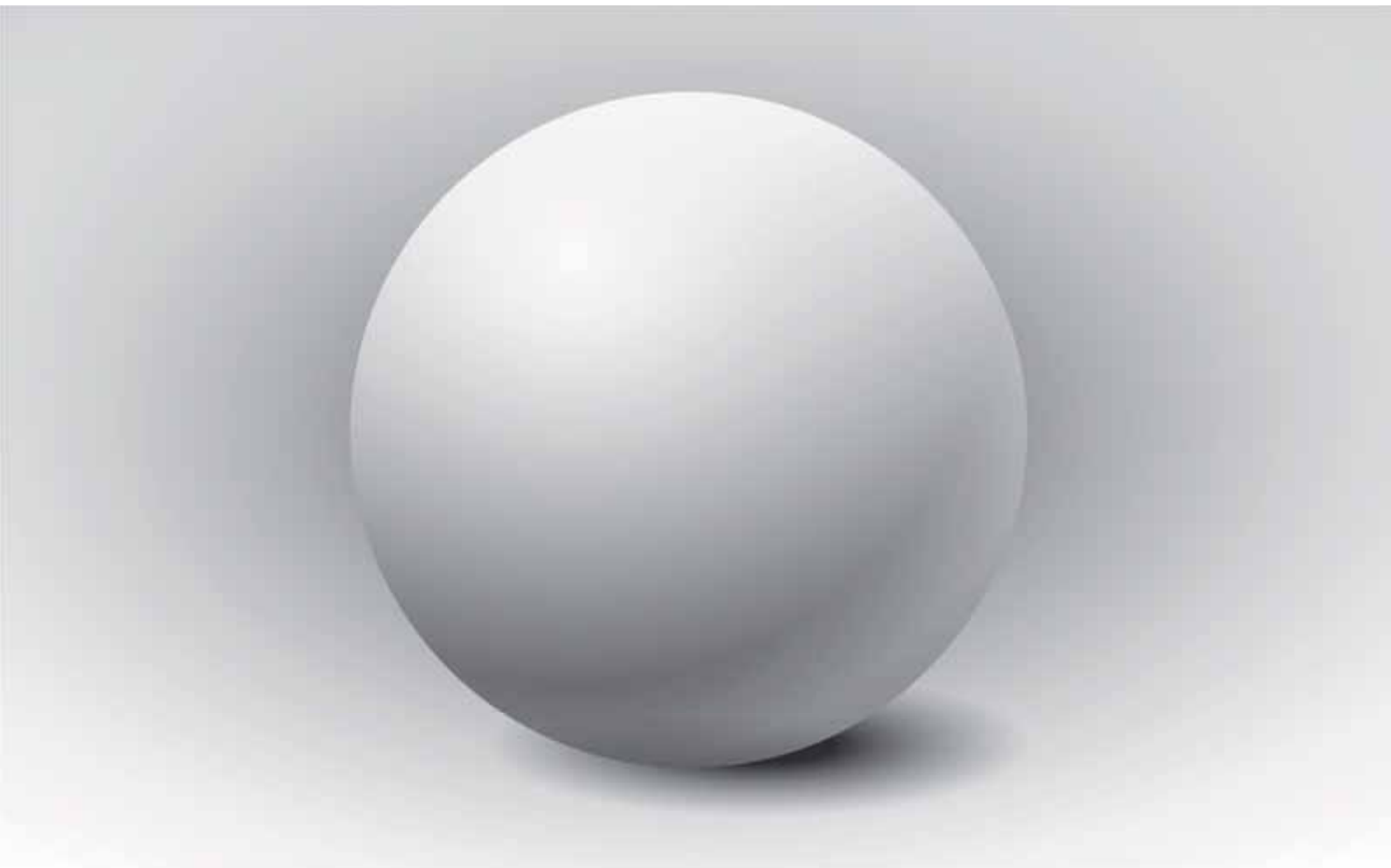
3. Делаем копию первого градиента. Для этого выделяем пункт в списке **Fill** и нажимаем **Cmd + D**. Таким образом на фигуре квадрата теперь два слоя заливки. Похожим образом работают слои и для обводки **Stroke**.

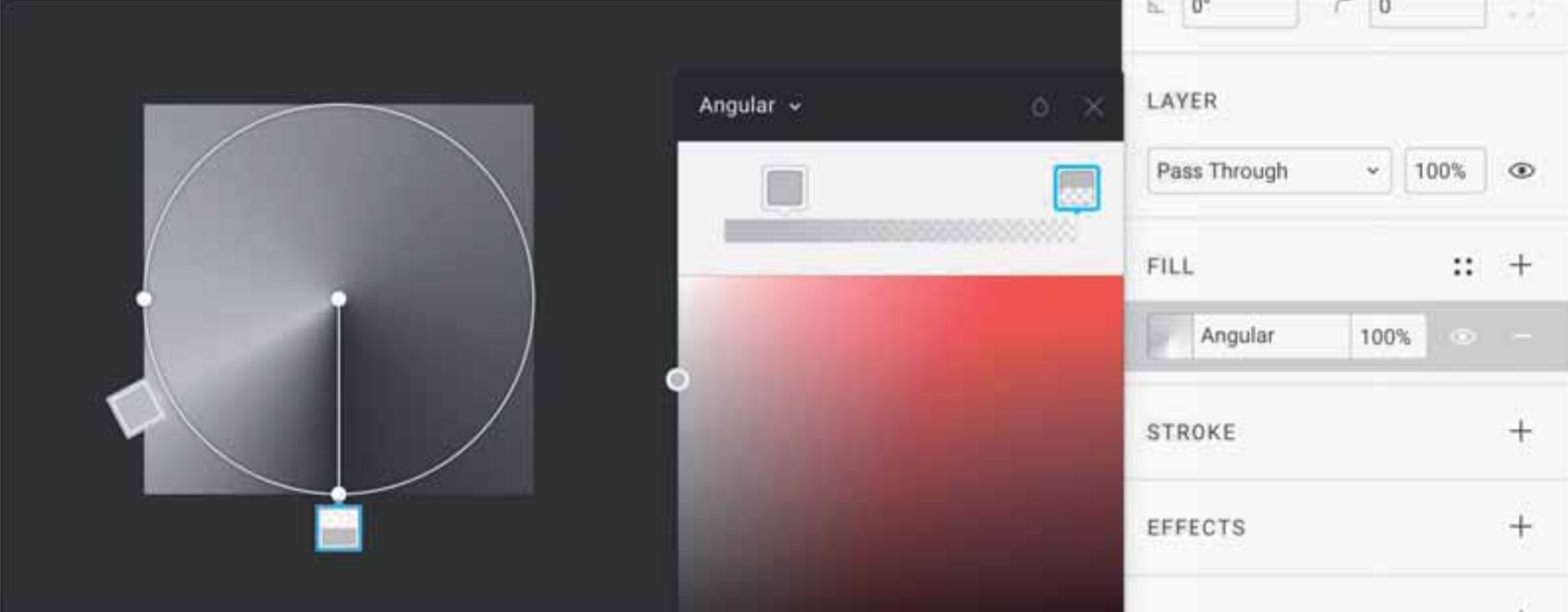


4. Смещаем второму градиенту угол и двигаем перпендикуляр как можно ближе к центру.
5. Ставим иконку капли справа — режим наложения этого градиента в режим **Color Dodge**. Получается эффект блика.

Рисование округлых поверхностей

При помощи радиальных градиентов можно прорисовывать округлые поверхности, имитируя объём.





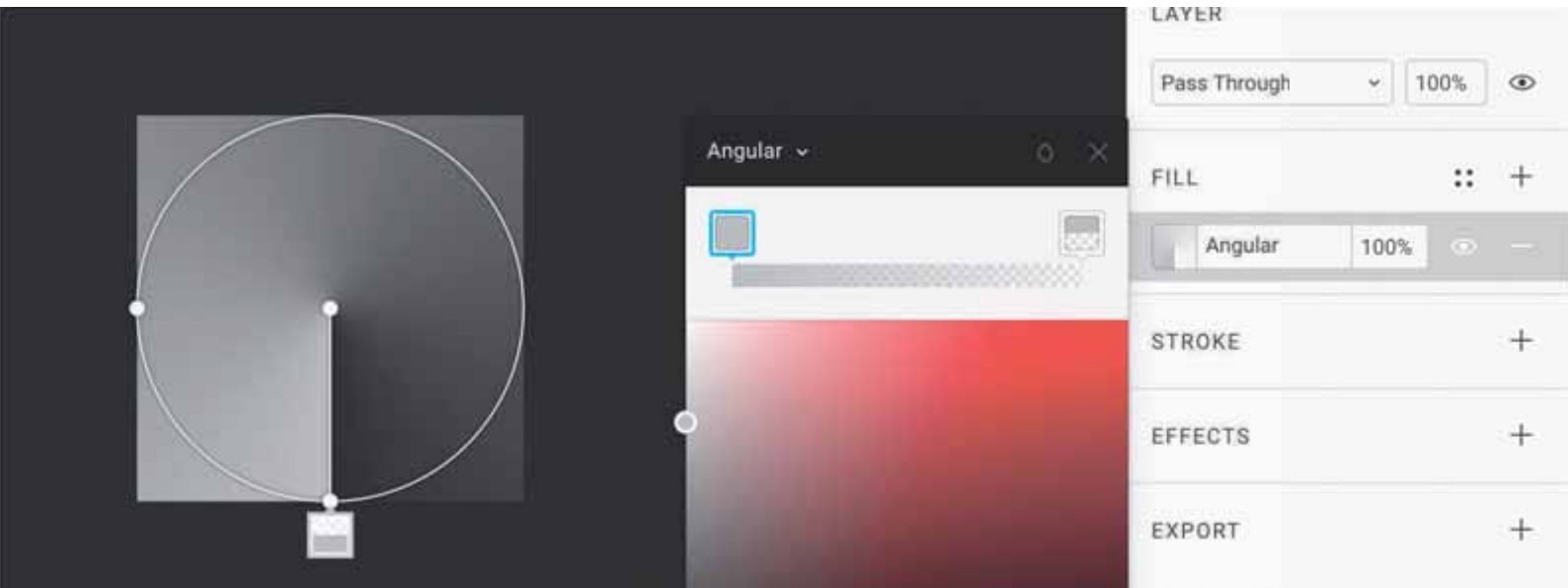
Режим заливки Angular: угловой градиент

Такой градиент формируется на основе угла в 360° , в котором переход цвета идёт от начальной точки до конечной. Поскольку этот угол составляет полный круг, для управления таким градиентом тоже используется эллипс. Однако, переход цвета в нём происходит по внешнему контуру, а не по радиусу.

Помни, что угловые градиенты не поддерживаются в формате SVG, поэтому такие фигуры придётся экспортировать в растр.

Задача: Сделаем простой радиальный градиент.

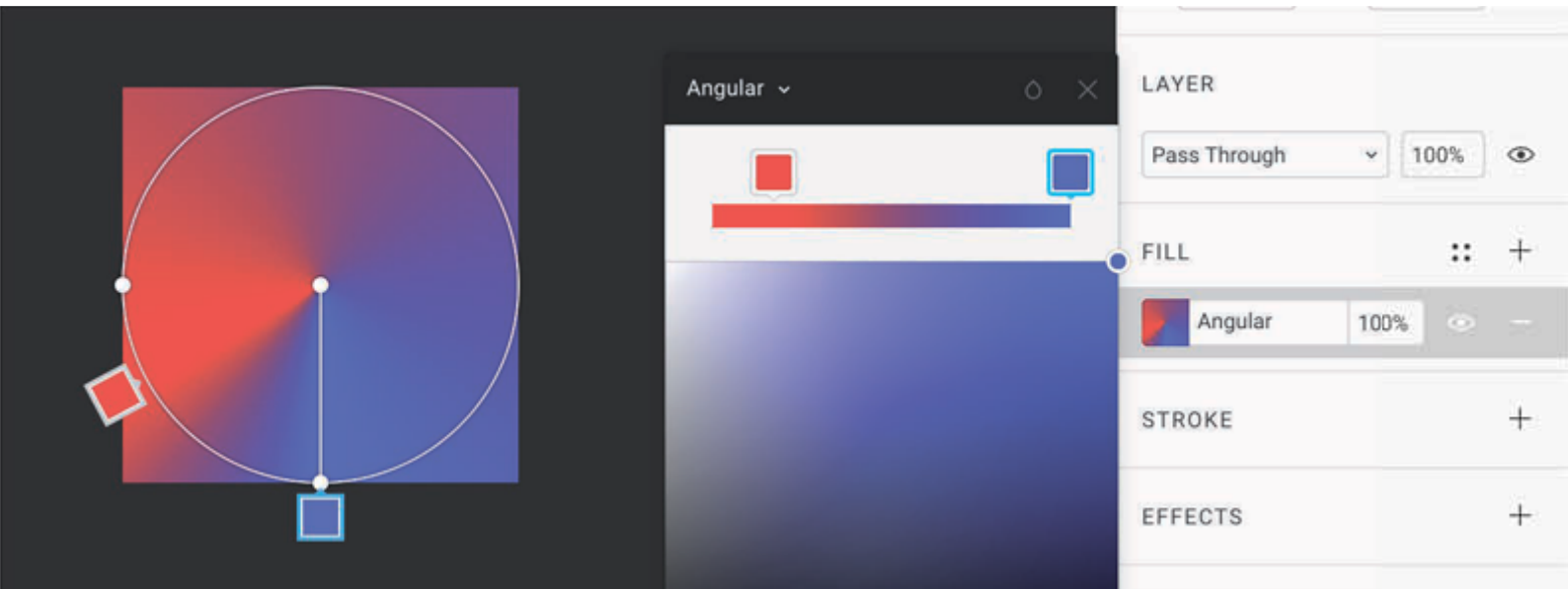
1. Создадим квадрат, **R**.
2. Переключим заливку в режим **Angular**. Будет видна окружность, диаметр которой соответствует длине квадрата.



Начальная и конечная точка стоят на 6 часов, а переход происходит по часовой стрелке. Как и в любом другом режиме, градиент изначально состоит из одного цвета с разной насыщенностью.

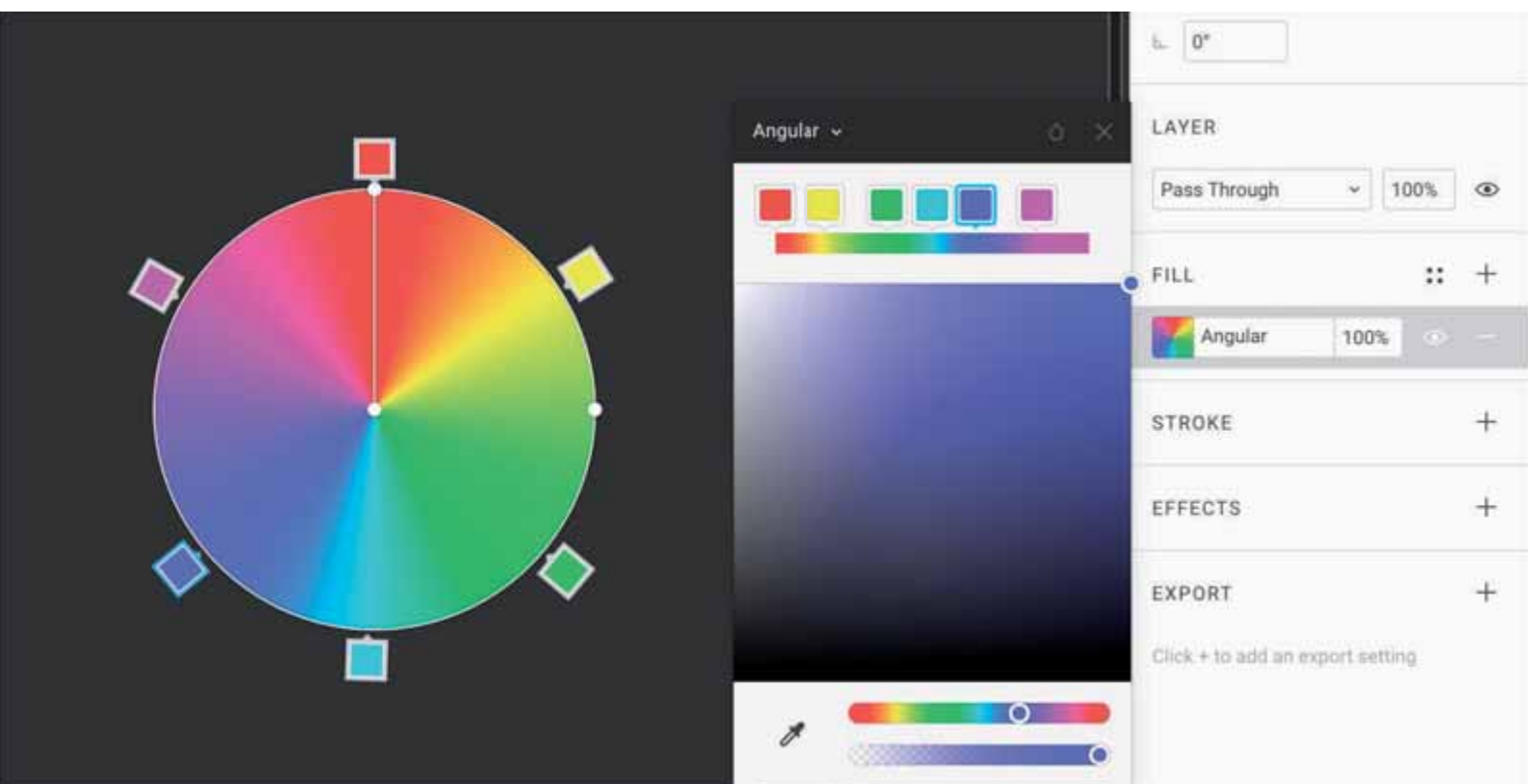
Начинается с полностью видимого цвета #C4C4C4, а заканчивается полностью невидимым. Поскольку задан тёмно-серый цвет холста, кажется, что градиент идёт от светлого к тёмному, но на самом деле один и тот же серый лишь теряет насыщенность.

3. Чтобы устройство углового градиента стало нагляднее, перетащим начальную контрольную точку цвета налево, на 8 часов. Конечная точка так и останется на 6 часов. Теперь они не накладываются друг на друга.

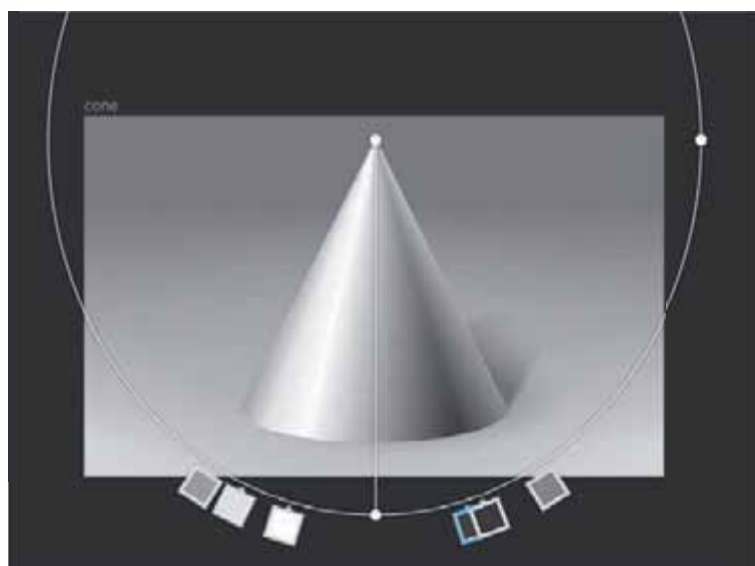
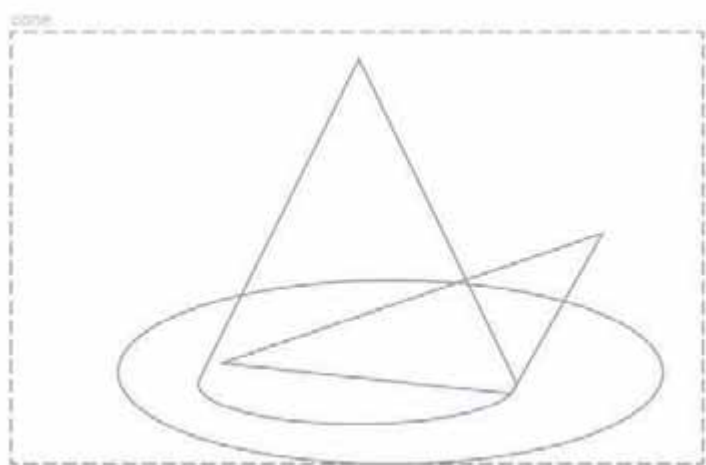


4. Меняем цвета на более различные: первая получит красный, а вторая синий. Теперь явно видно, как в обе стороны от красной точки цвет равномерно переходит от красного к синему.
5. Также как и в других типах градиентов можно задавать контрольные точки на шкале слева.

Используя угловой градиент, можно сделать круговой спектр, разделив окружность на 6 равных отрезков:

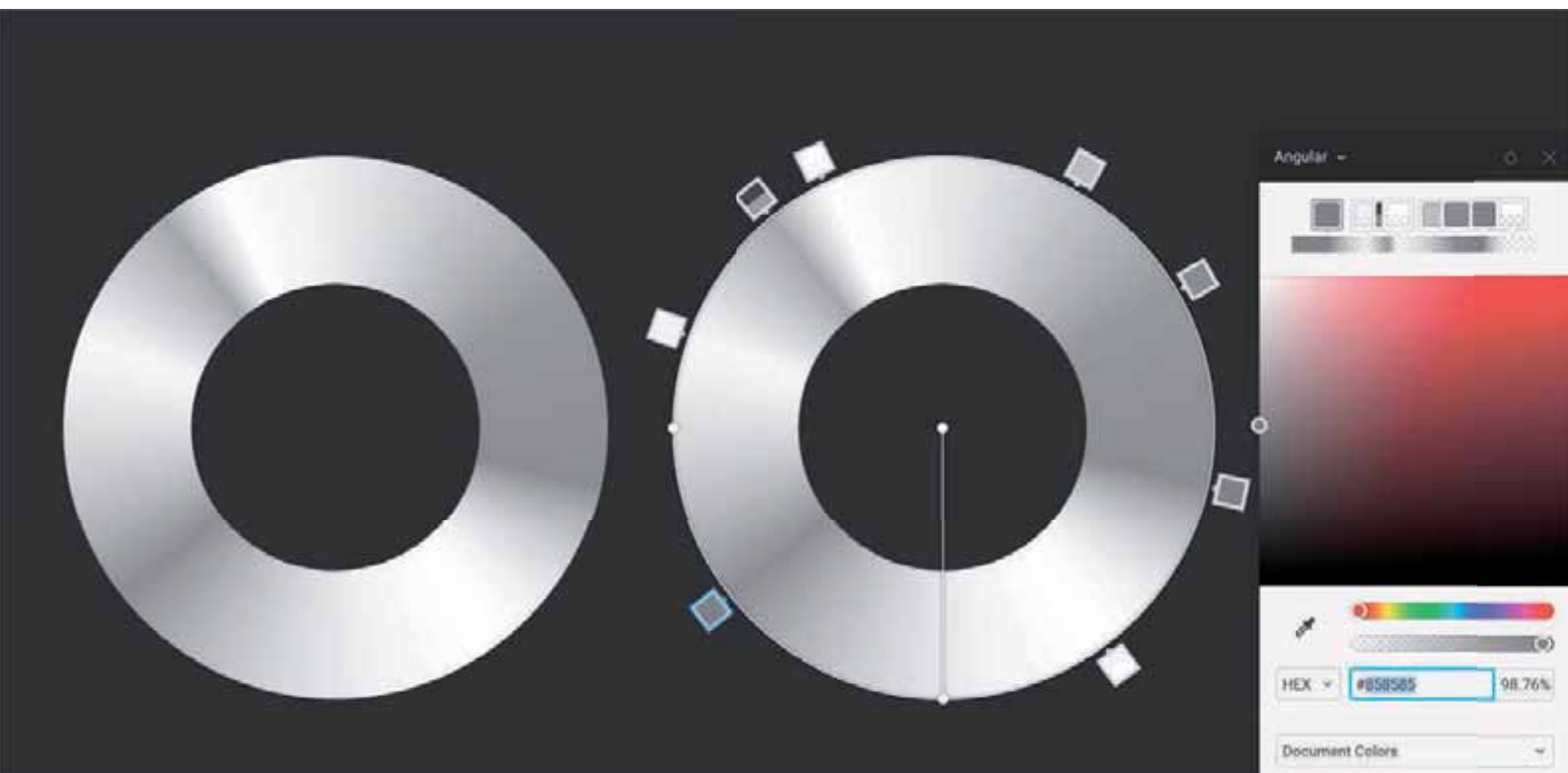


Также при помощи углового градиента можно нарисовать конические поверхности. Саму форму конуса можно нарисовать при помощи эллипса и пера. Затем, объединить их в одну фигуру, разорвав верхнюю точку эллипса.



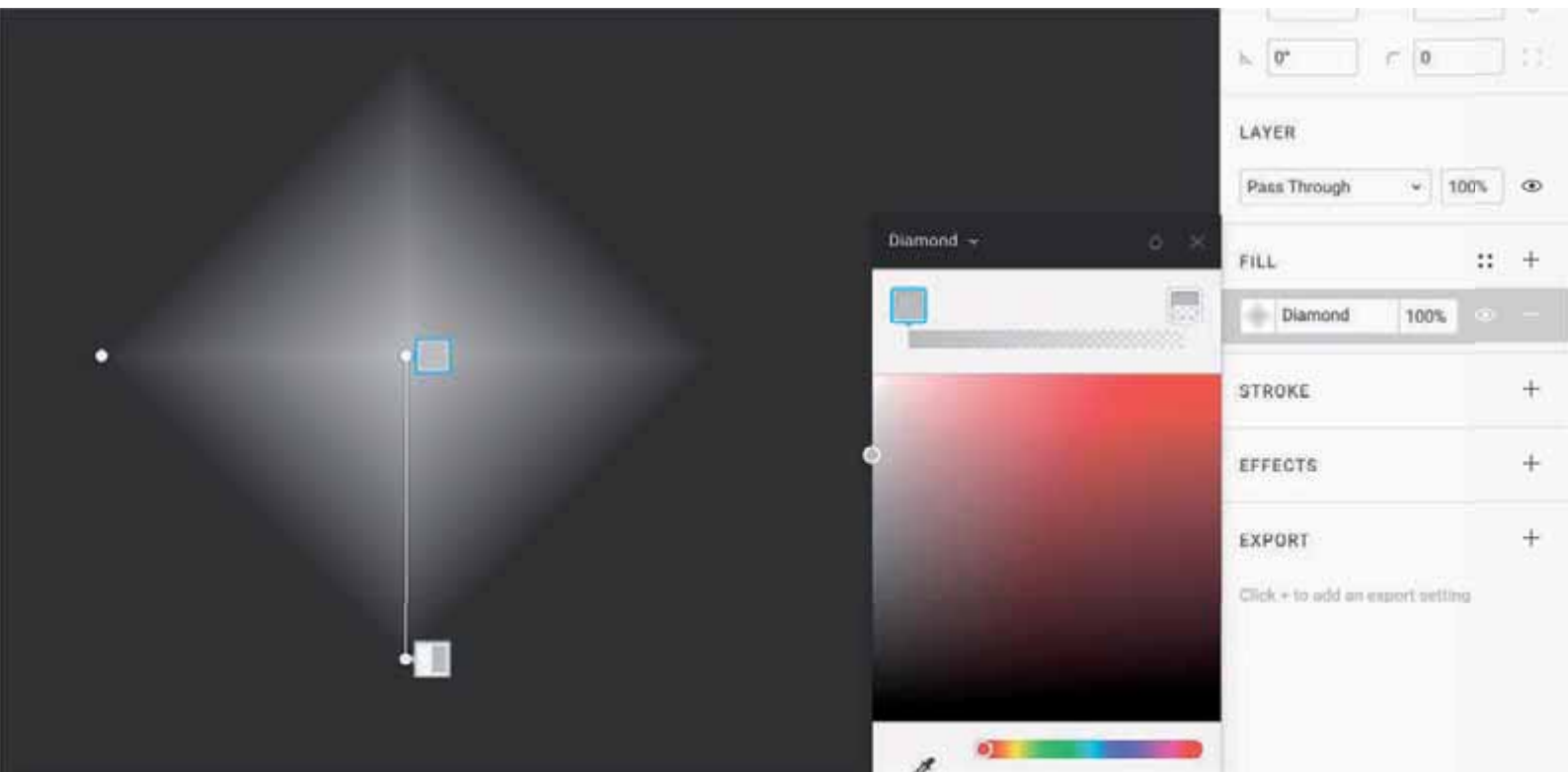
Ещё одно применение радиальных градиентов — круглые поверхности, похожие на хромированные. Из них можно накрутить ручки, кнопки и прочий скевоморфичный UI.

~~Только, пожалуйста, не надо, это уже не модно.~~



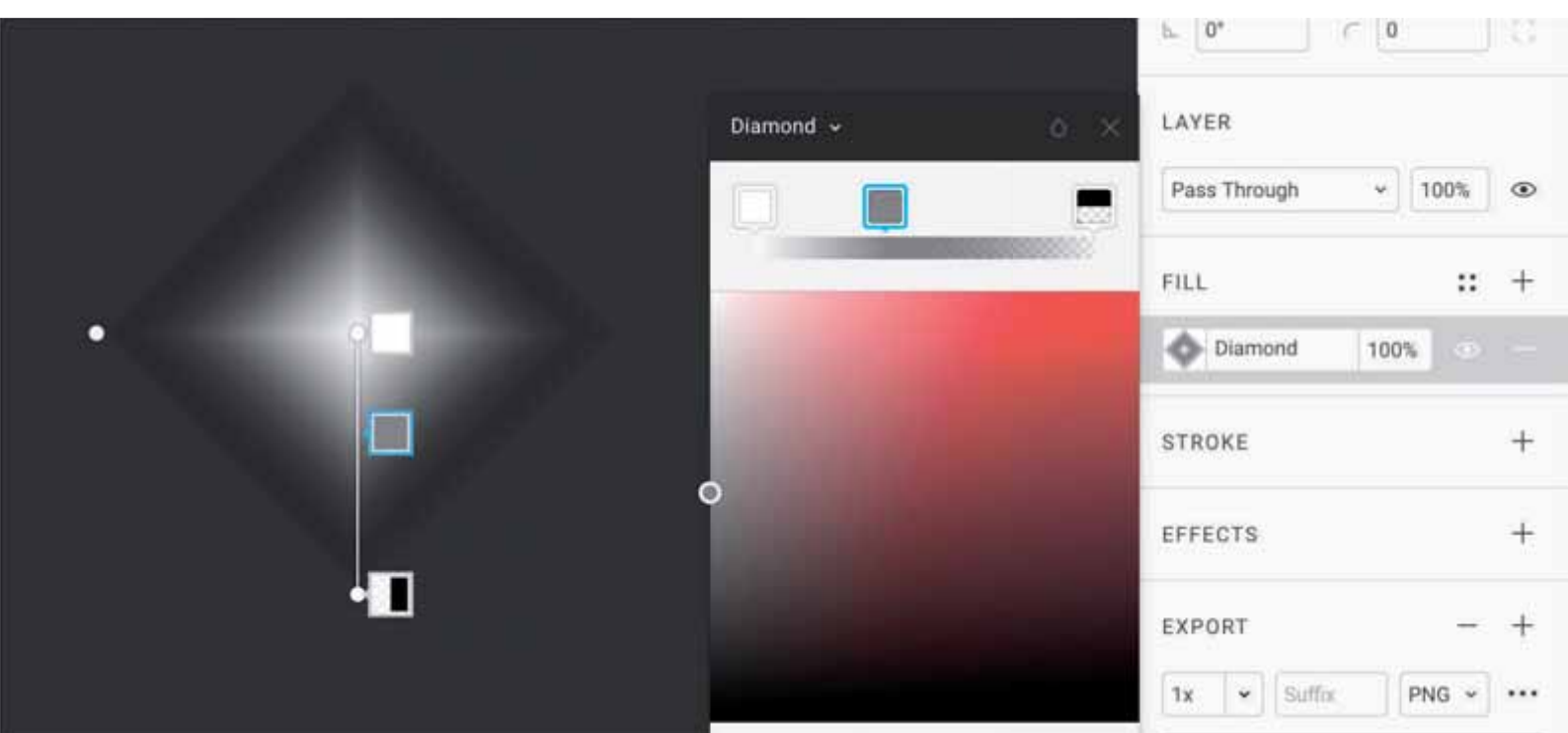
Режим заливки Diamond: делаем блики

Новый режим, которого нет в Скетче. Позволяет делать градиенты, похожие на радиальные, с тем отличием, что внешний край перехода имеет форму ромба.

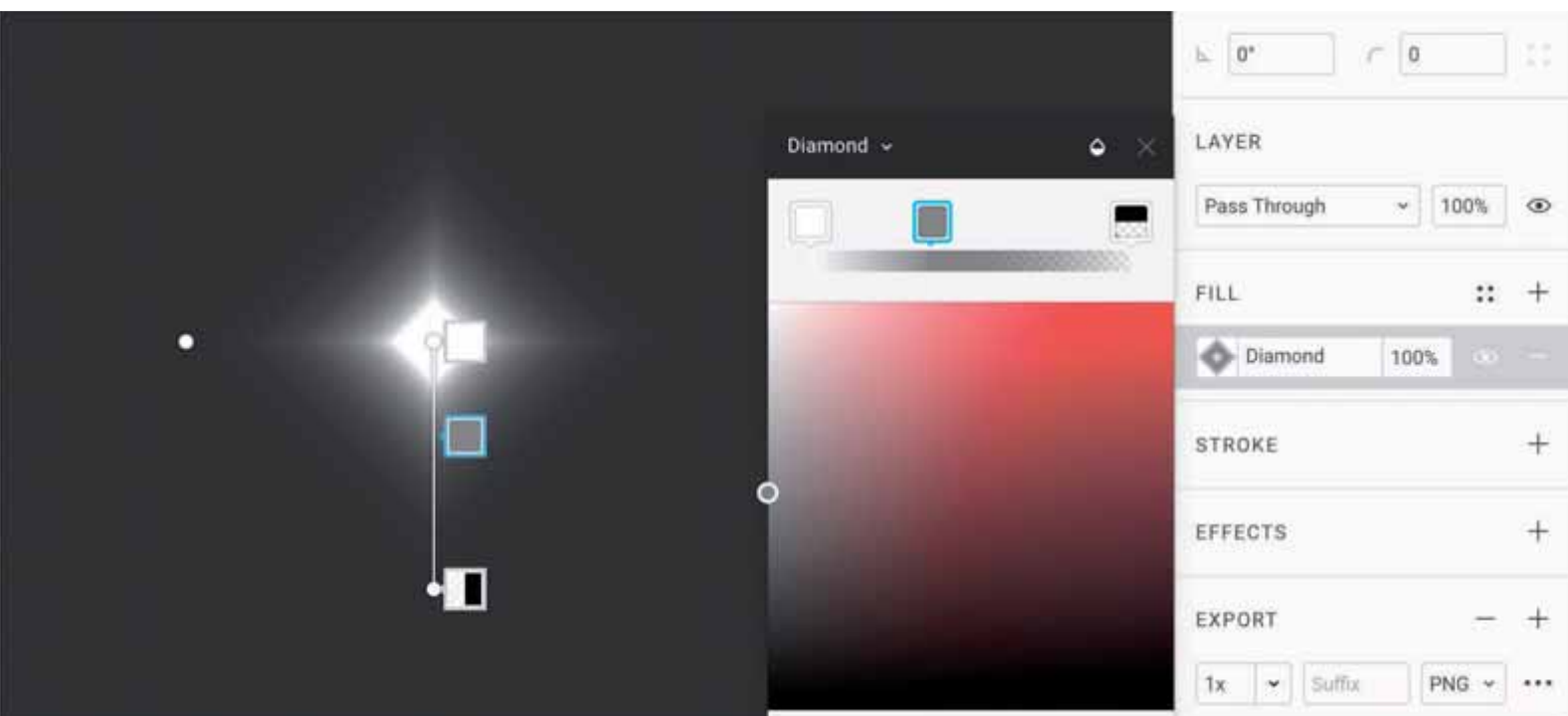


Применяется редко. Годен в основном для бликов на нелепых хромированных ретро-футуристических логотипах в стиле 80-х. Но даже в них следует осторожничать с силой блика, иначе получится нереалистично и дёшево.

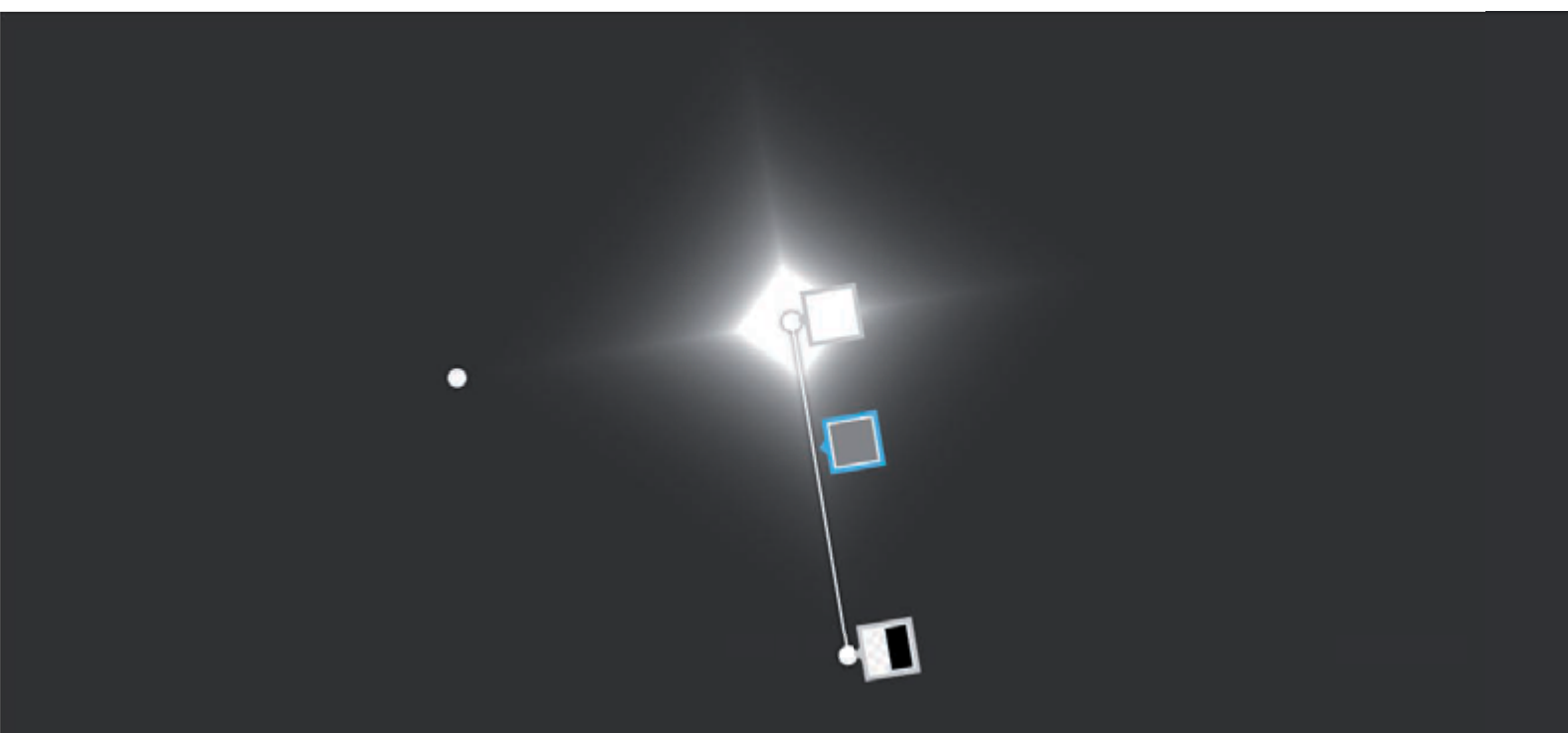
1. Создадим квадрат, **R**.
2. Переведём его заливку **Fill** в режим **Diamond**.
Как и в радиальном градиенте, в этом типе ось перехода цвета начинается от центра фигуры и заканчивается на внешней границе.
3. Выберем первым цветом белый, посередине шкалы поставим серый, а в конце — чёрный. Полностью уберём ему опасность.



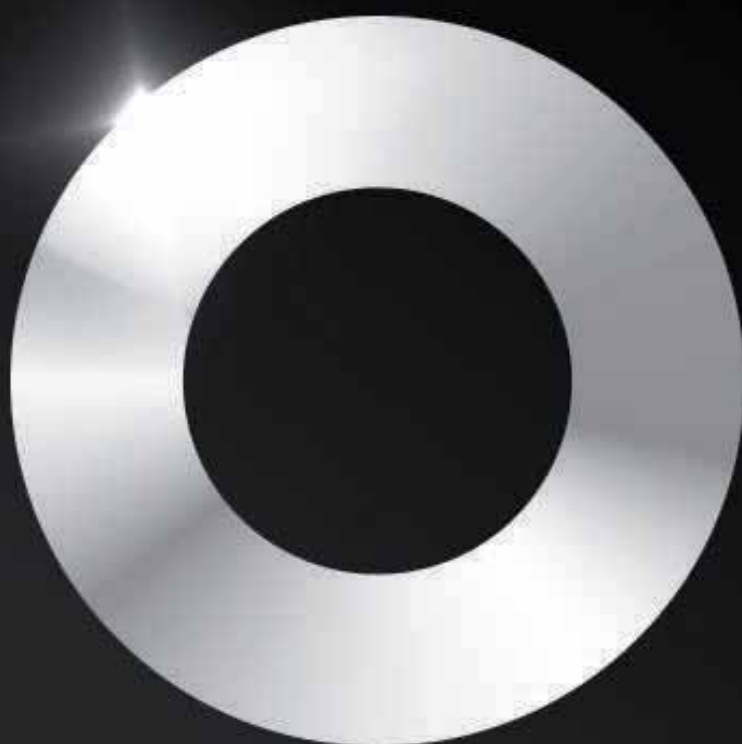
4. Переключим градиенту режим наложения (меню с каплей) в **Color Dodge**. Этот режим усиливает белый в центре и приглушает серый на краях, давая ощущение пересвета:



Двигая серую точку, можно настроить желаемый размер белого засвеченного ромба. Желательно, чтобы форма внешнего ромба вообще была размыта. Останется только внутренний белый ромб и четыре луча.



5. Загибаем угол, держась за нижнюю точку.
6. Выходим из редактирования градиента, **Enter**.
7. Ставим блик на край фигуры, где мог бы отражаться самый яркий источник света.

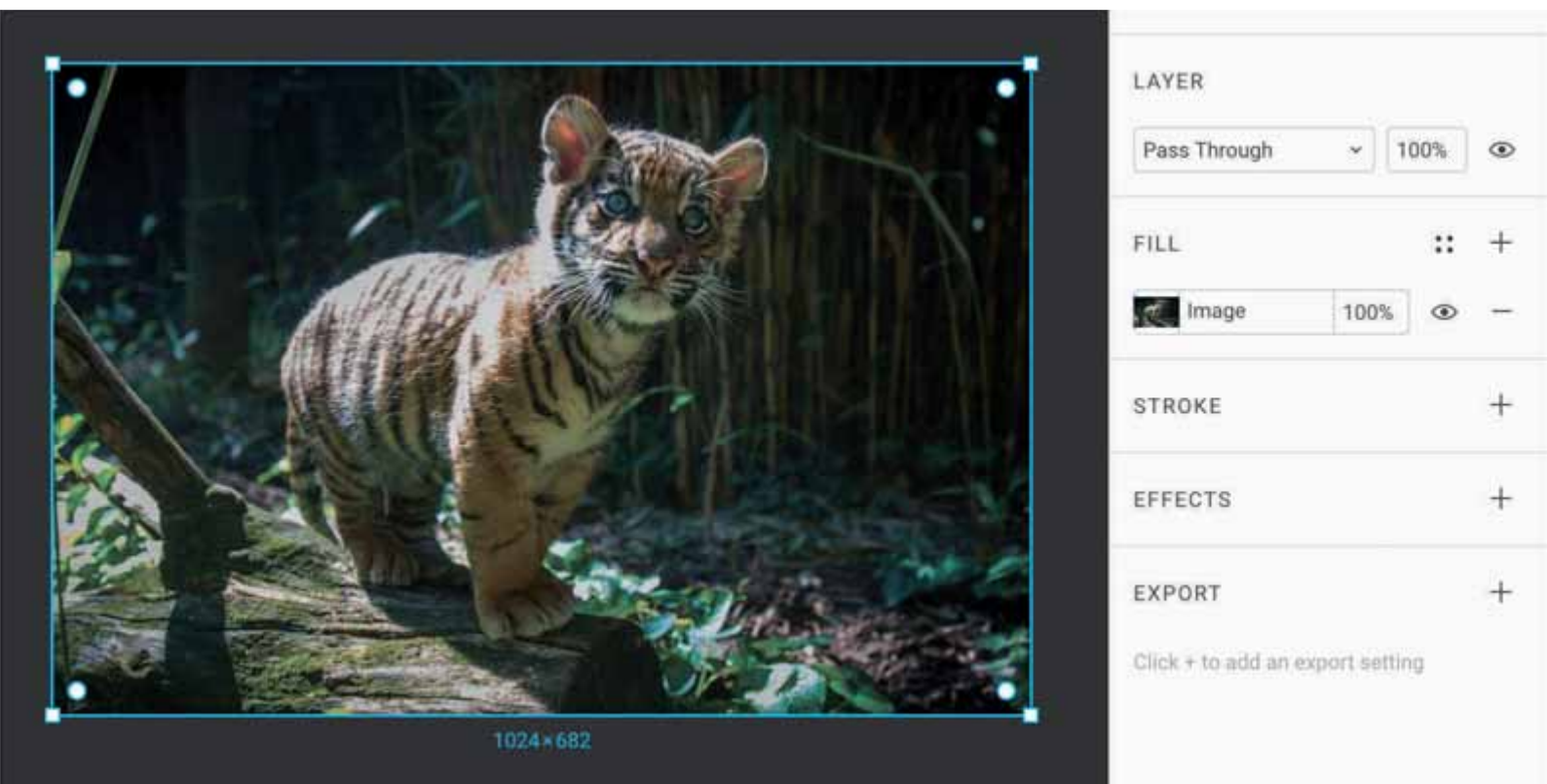


15. Режим заливки Image

[Проект в Фигме](#) →

Режим **Image** — последний из режимов в списке, но один из важнейших. Позволяет накладывать на фигуры растровую заливку: фотографии, паттерны и текстуры.

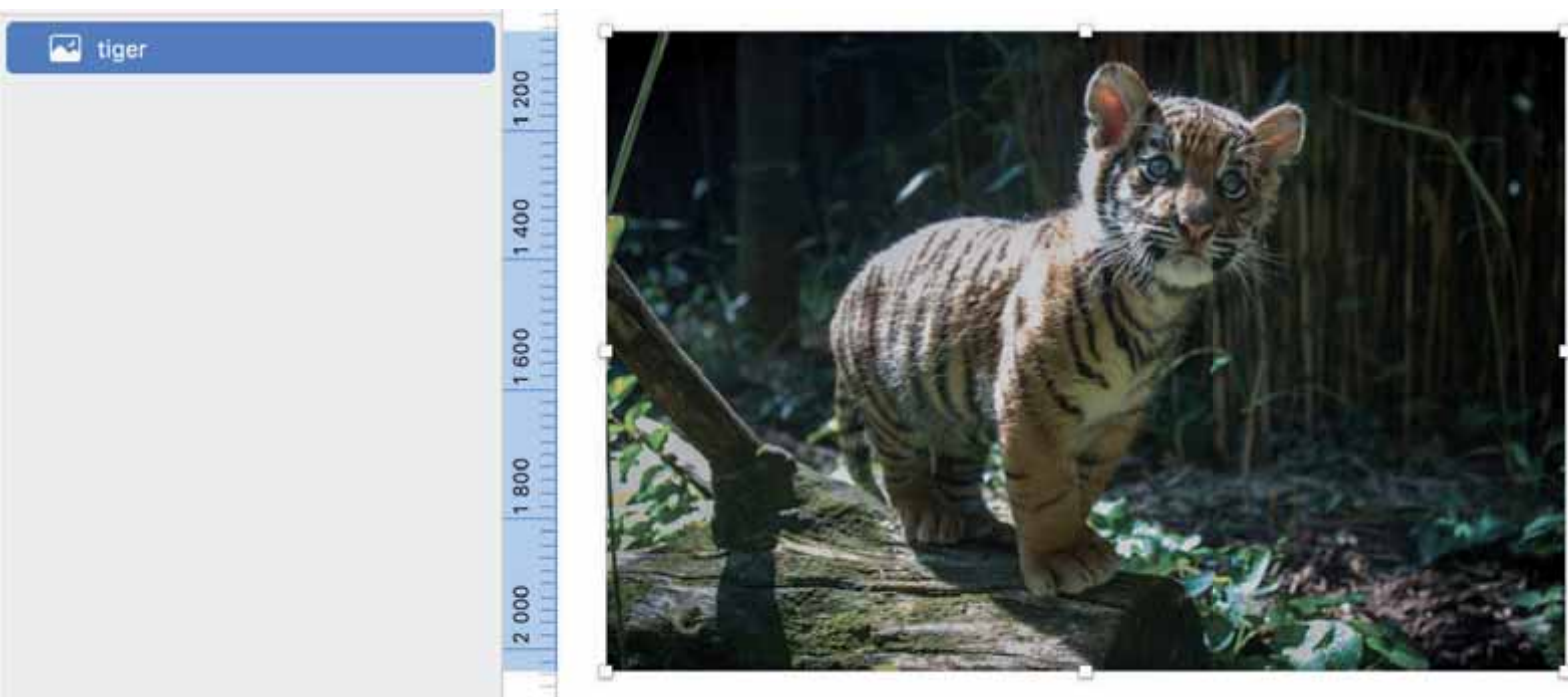
Важно понять эту концепцию. Если в Фигму вставить любой JPG или PNG-файл, он появится на холсте как векторный прямоугольник. Растровое изображение будет наложено на него в режиме **Image**. Вставляем растровый файл в Фигму. Это можно сделать перетаскиванием, или командой **Place Image**, **Shift + Cmd + K**.



Источник фото: Roshan Patel, [Smithsonian's National Zoo](#)

В отличие от всех остальных режимов заливок, если выделить слой с растром и нажать **Enter**, мы перейдём в панель работы с заливкой, а не в режим редактирования векторной фигуры. Так интерфейс расставляет акценты, что в режиме **Image** пользователю важнее работать с заливкой, а не с шейпом.

В Скетче есть отдельный тип объектов — **Bitmap**. Внимание на иконку слева. Если вставить растровый файл, он появится на холсте как битмап, и в этом ключевое отличие Скетча от Фигмы при работе с растром.



Битмапы можно обрезать и заливать цветом. В Скетче тоже можно наложить на прямоугольник растровую заливку, однако, мы не сможем её редактировать и как угодно кадрировать после вставки.

Поскольку в Фигме нет битмапов, мы редактируем растровые объекты только через заливку.

Выделяем слой с фото и открываем панель заливки, **Enter**.

Сам режим заливки **Image** может быть в 4 вариантах.

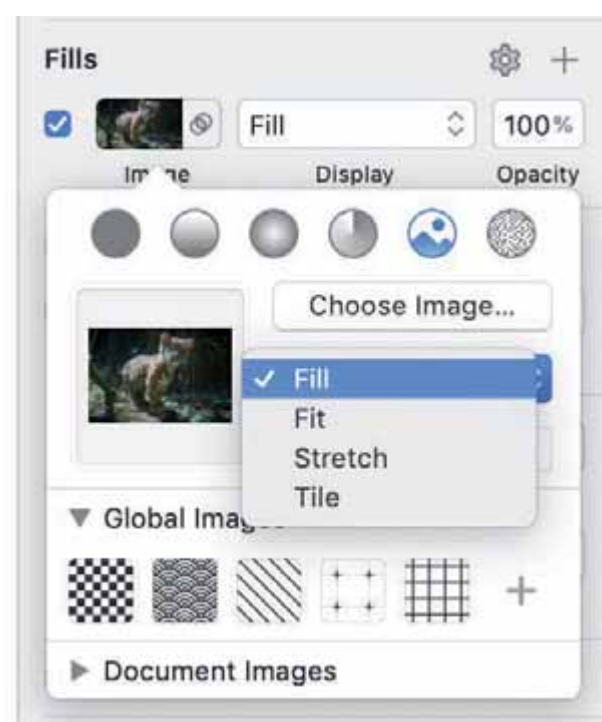


- **Fill** — вся площадь векторной рамки заполняется растром.
- **Fit** — растр вписан в векторную рамку.
- **Crop** — режим, в котором можно тонко настроить, как нужно кадрировать растр, меняя форму векторной рамки, наклоняя или искажая пропорции растра.
- **Tile** — режим, в котором растр используется как паттерн и заполняет собой площадь фигуры словно плитка.

В Скетче режимы почти такие же:

- **Fill**
- **Fit**
- **Stretch** — заполнение фигуры растром с искажением пропорций
- **Tile**

Режима **Crop** нет, поскольку эта логика реализуется через обрез битмапов.



Режим Image / Fill

Установлен по умолчанию. Действует как заливка растром в Скетче в одноимённом режиме. Растровое изображение максимально заполняет площадь фигуры. Центр растра в той же точке, что и центр фигуры-рамки. Если растянуть прямоугольник, растр подстроится по большей из сторон. Потянем ширину:



Растр заполнит всю доступную ширину. Фигма скроет излишки сверху и снизу.

Если ужимать ширину так, что она станет меньше высоты, растр будет заполнять всю доступную высоту.



Центр растра и прямоугольника по-прежнему остаются в одной точке. Сдвинуть изображение внутри прямоугольника, сместив центральную точку, нельзя. Для этого используем режим **Crop**.

Режим Image / Fit

В этом режиме растр вписывается в фигуру по меньшей из сторон: он всегда виден и не обрезается прямоугольником.

Если у прямоугольника и растра в нём разные пропорции, по бокам появляются пустоты:

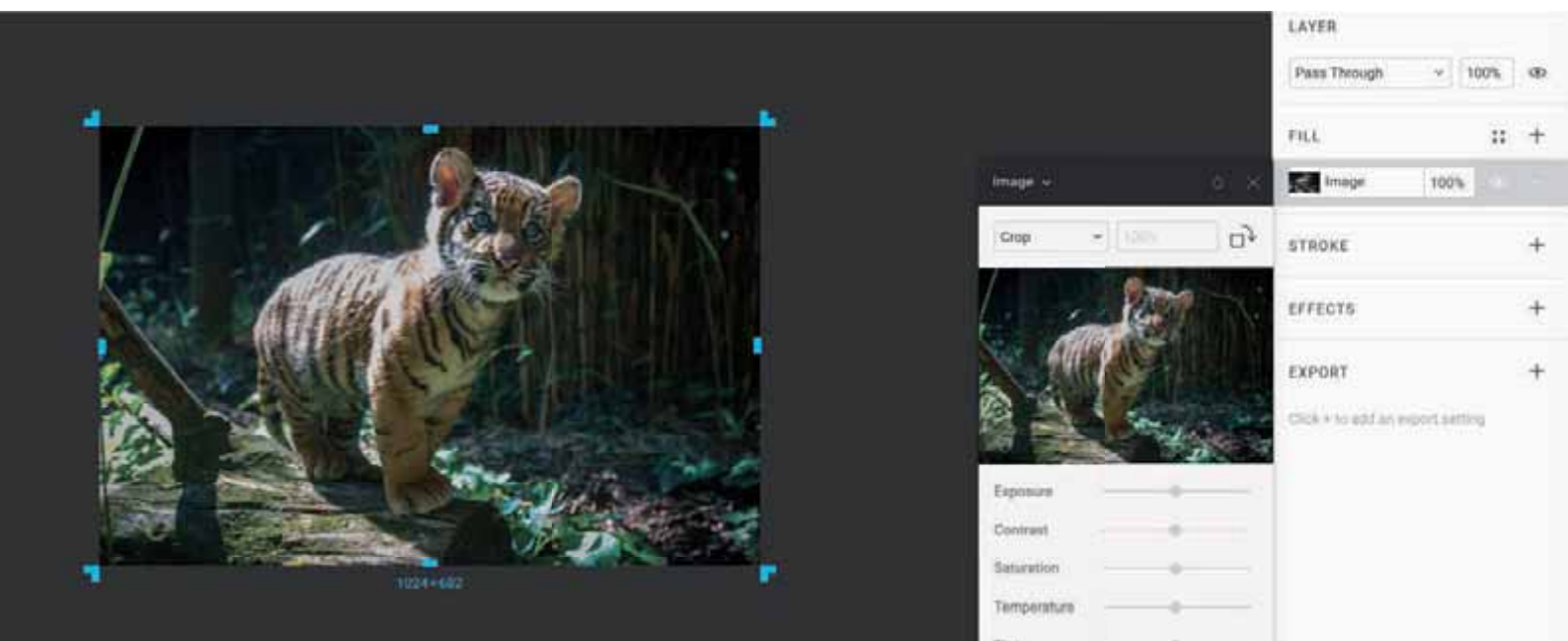


Режим Image / Crop

Когда мы вставляем изображение в Фигму, очень часто нужно его обрезать. Режим **Crop** позволяет менять пропорции векторной рамки, в которой есть растровая заливка. При этом, растр не смещается.

Если мы хотим задать точный размер обрезанного изображения в пикселях, это лучше делать в режиме **Fill**. Если в режиме **Crop** задавать значения ширины и высоты рамки с клавиатуры, растр будет искажаться.

1. Переключаемся на режим **Crop**.

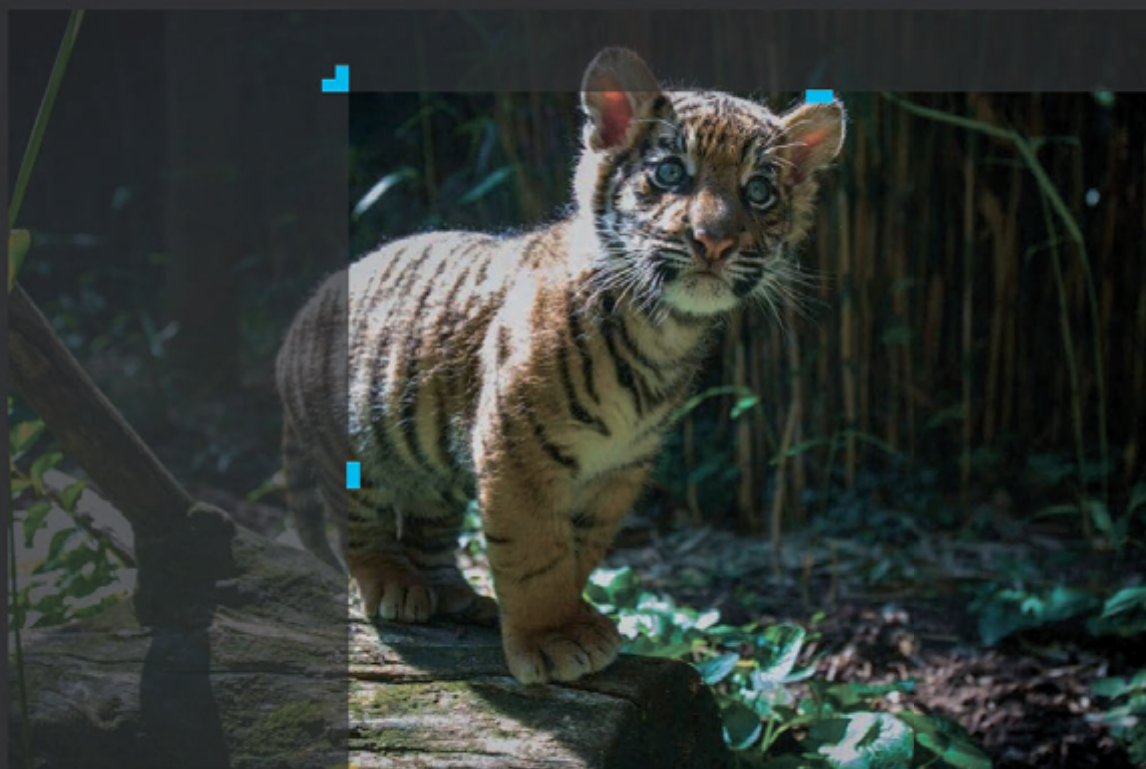


2. Тянем за любую ручку к центру изображения. Рамка сжимается.



Чтобы кадрировать от центральной точки, зажимаем **Opt**.

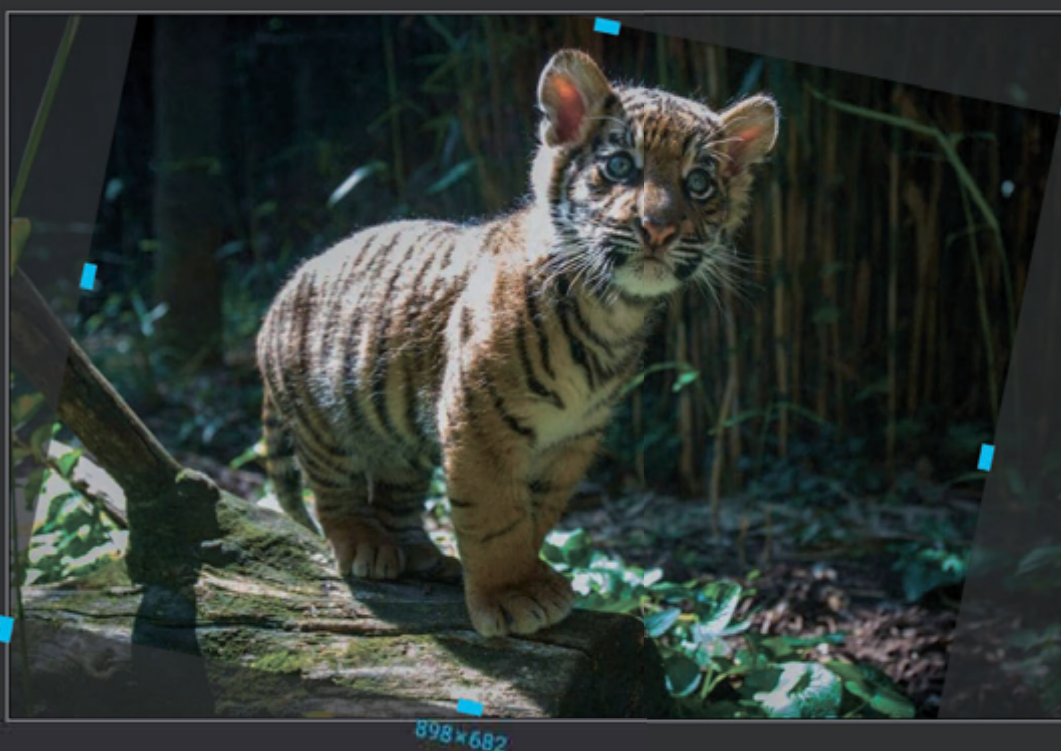
3. Также в этом режиме можно сдвигать изображение в рамке.
Для этого зажимаем **Cmd**, берём за растр и тянем его в нужную сторону:



842×682

4. Рамку кропа можно наклонять и таким образом исправлять заваленный горизонт. Для этого наведём на любую из угловых ручек, пока указатель не превратится в изогнутую стрелку.

Наклоняем кроп-рамку:



5. Также можно наклонять и сам растр, держась за его угол.



Если необходимо исказить его пропорции, тянем за белые края.

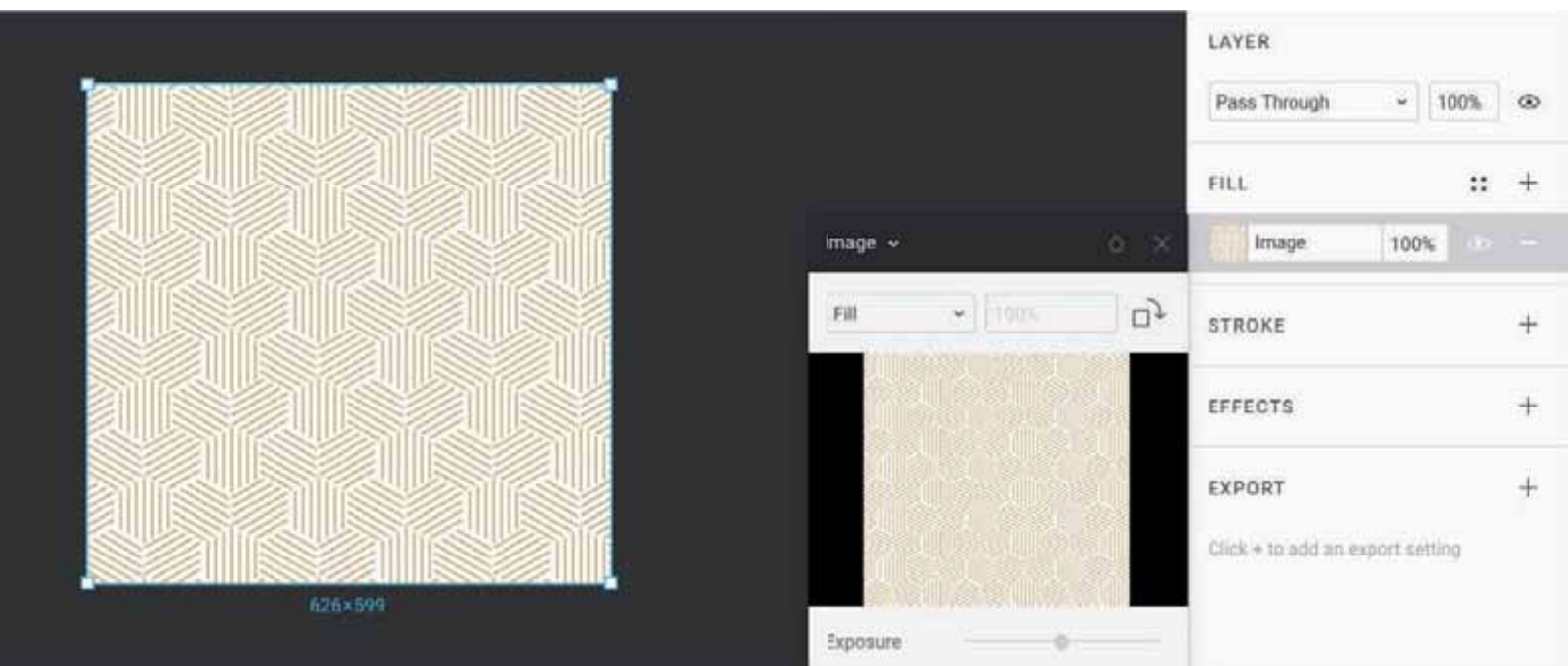
6. Чтобы зафиксировать смещённое или обрезанное изображение, дважды кликаем в пределах рамки или нажимаем **Enter**. При этом исходное изображение не меняется, а лишь маскируется. К нему всегда можно вернуться, выделив слой и нажав **Enter** ещё раз.

Чтобы сбросить смещение и угол наклона кропа, можно вернуться в режим **Fill**.

Режим Image / Tile

Режим **Tile** [тайл] используется для заполнения фигуры повторяемыми фрагментами — бесшовными паттернами. Обычно они нарисованы так, что если их размножить и поставить вплотную друг к другу, швы между ними будут незаметны.

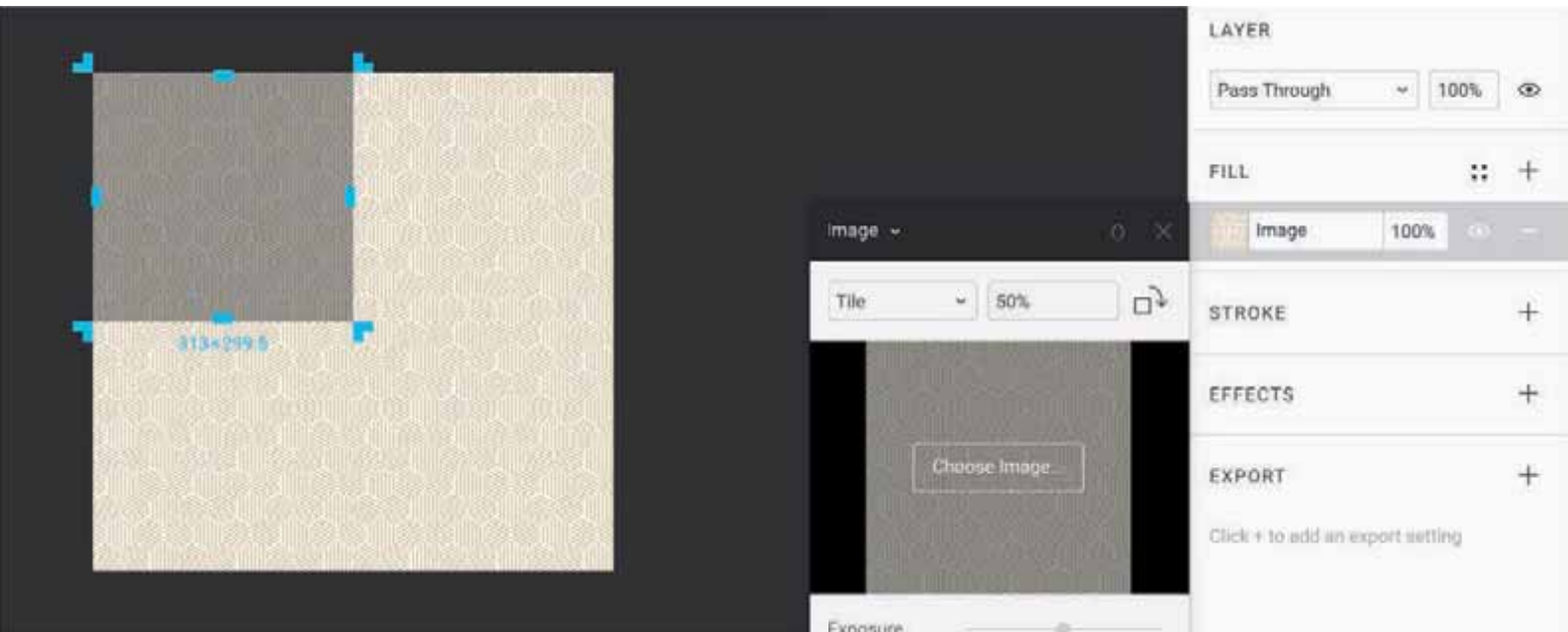
Вставляем растровый паттерн в Фигму:



Источник паттерна: [Vilmosvarga - freepik.com](https://www.freepik.com)

Переключаем режим **Image** с **Fill** на **Tile**.

Появляется тёмная рамка с ручками:



Она определяет, какого размера будет сетка, по которой паттерн замостит фигуру. Нижняя правая ручка позволяет настраивать желаемый размер плитки.

Вне зависимости от размера фигуры, паттерн будет распространяться на всю доступную площадь. Также его размер можно задавать в процентах в панели **Image**.



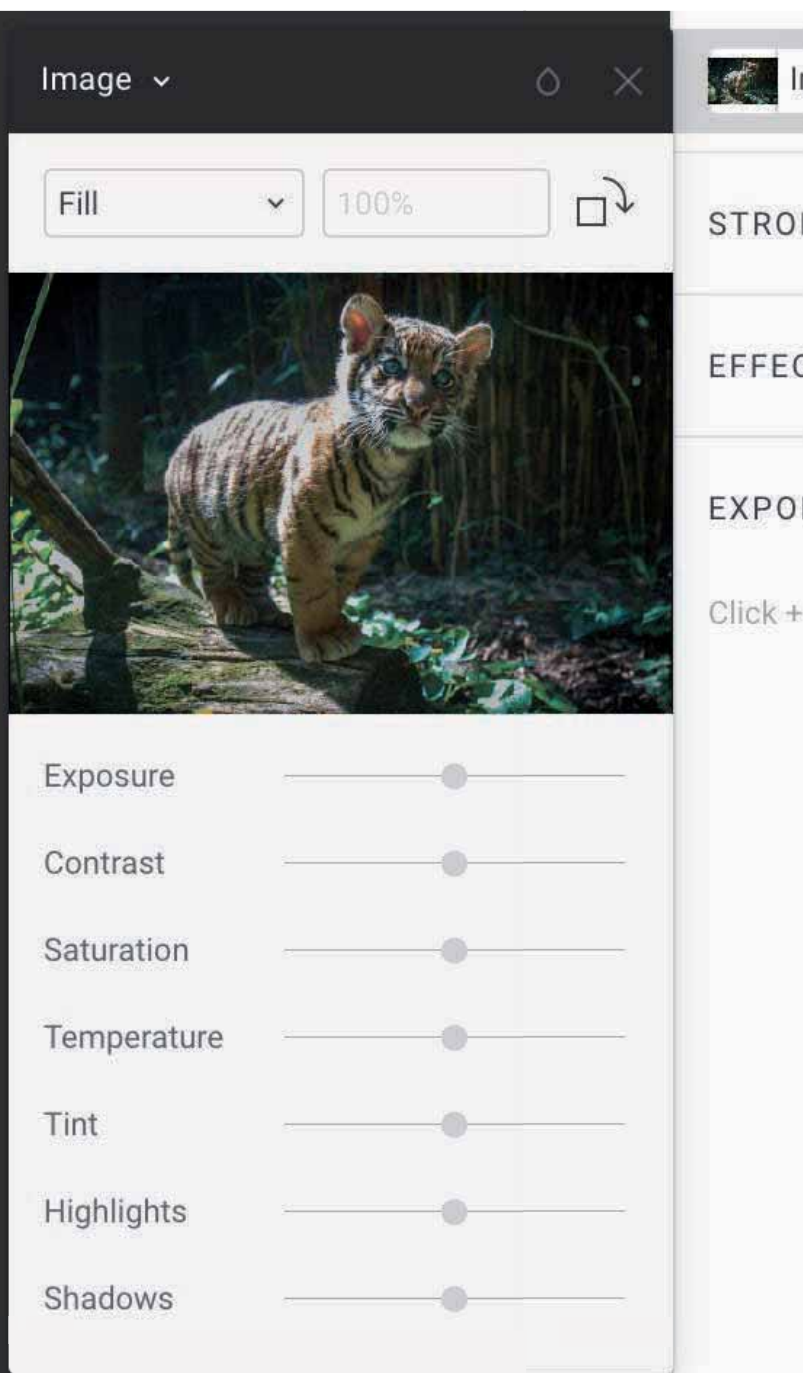
16. Цветокоррекция

[Проект в Фигме →](#)

Там только скриншоты.

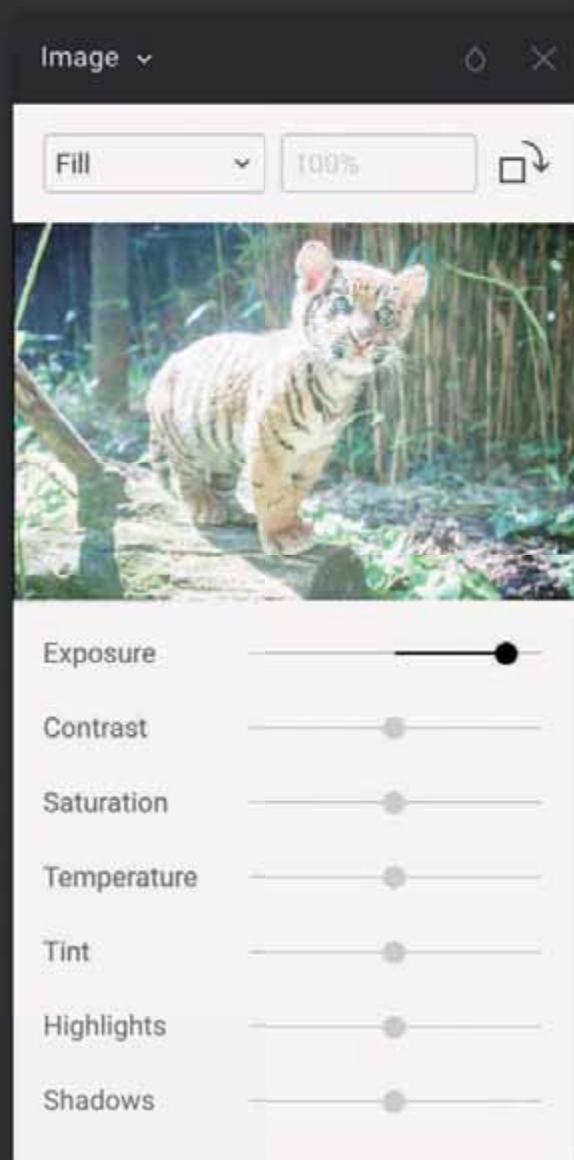
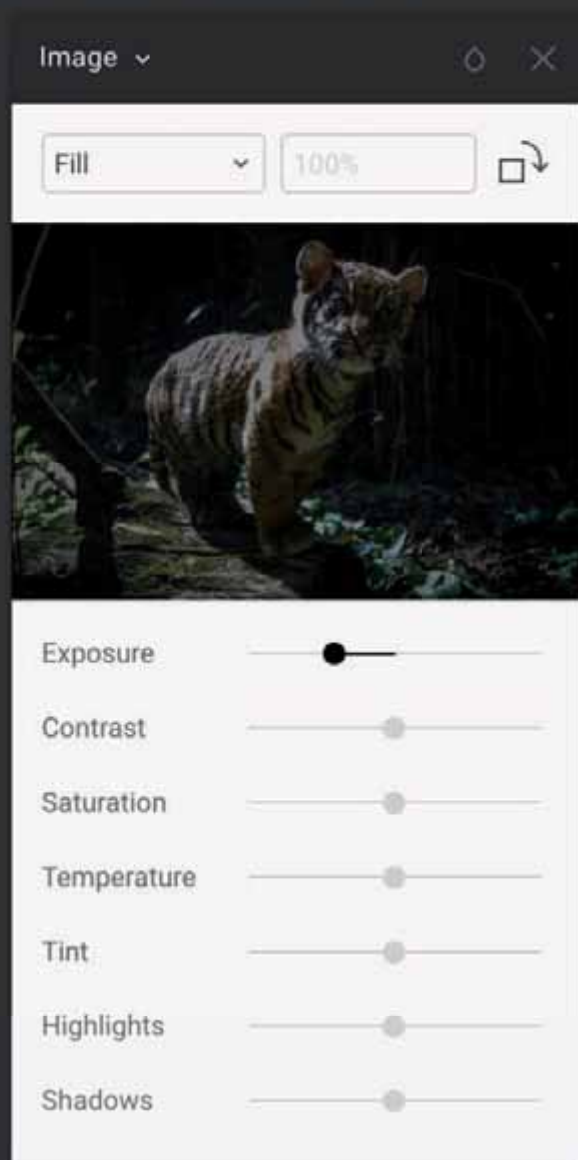
В Фигме доступны простые, но мощные фильтры для быстрой коррекции цвета фото без использования внешних редакторов.

Их вполне достаточно, чтобы сделать иллюстрации и аватары в дизайне поярче или придать им нужный характер. Все изменения коррекции не меняют исходное растровое изображение.



Корректировать цвета можно в панели **Fill** у тех слоёв, где есть растровая заливка. Если панель стоит в режиме Image, доступны следующие фейдеры:

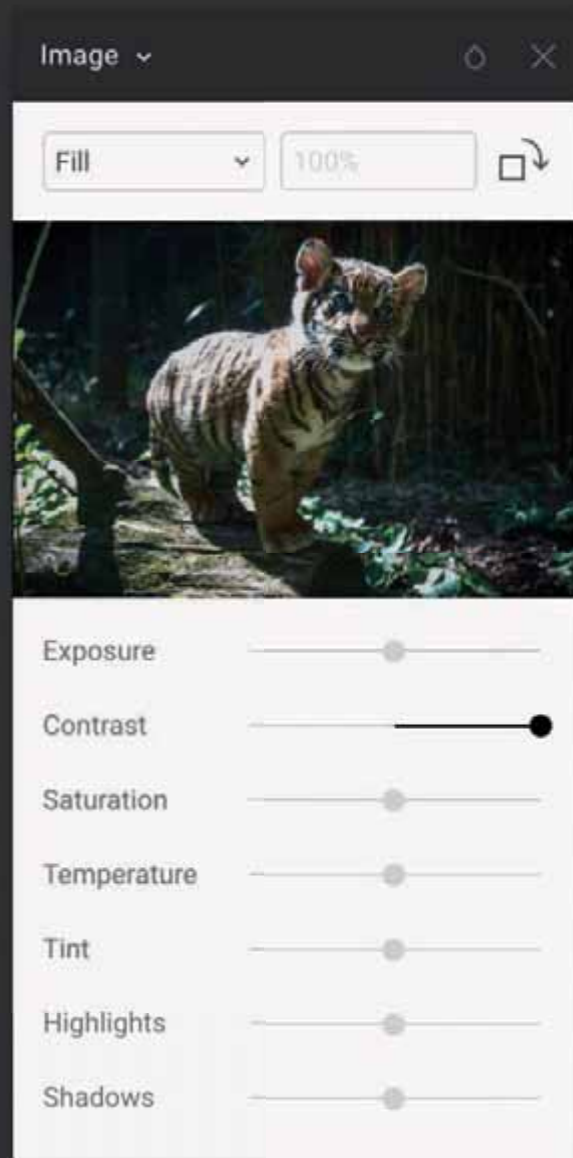
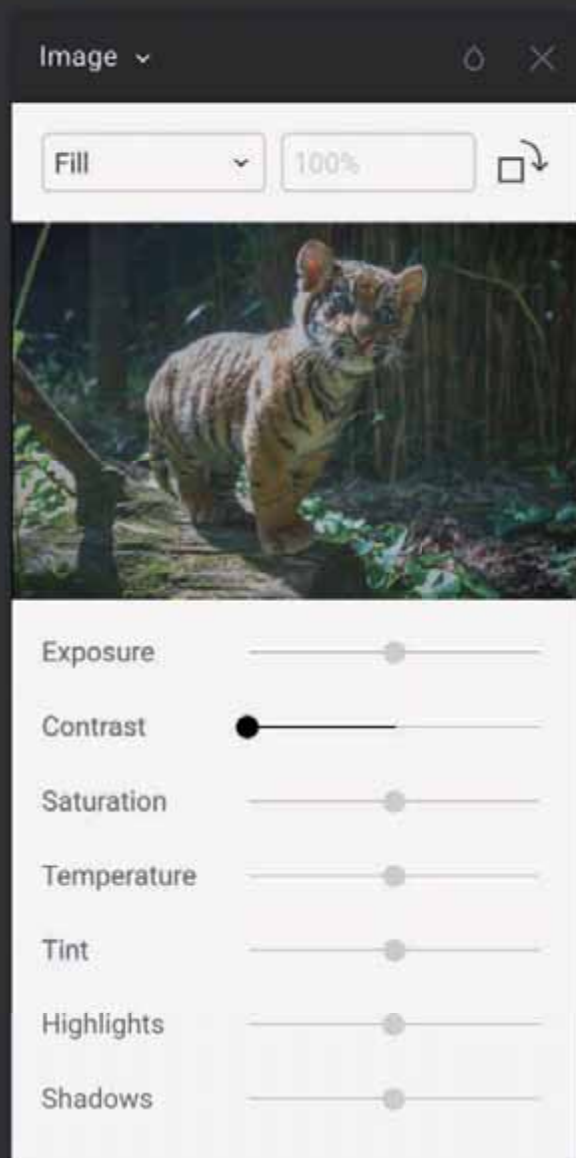
- **Exposure** [экспоужа] — экспозиция. Позволяет имитировать разное количество света в кадре. Темнее-ярче.
- **Contrast** — мягче-драматичнее.
- **Saturation** [сэтьюрэйшн] — насыщенность цвета. Если увести вправо, обесцветим.
- **Temperature** — температура света. Синий холоднее, жёлтый теплее.
- **Tint** — оттенок света, от зелёного до красного. Пригодится для имитации старых цветных фото.
- **Highlights** [хайлайтс] — коррекция светлых и пересвеченных участков.
- **Shadows** — коррекция тёмных и теневых участков.



Exposure

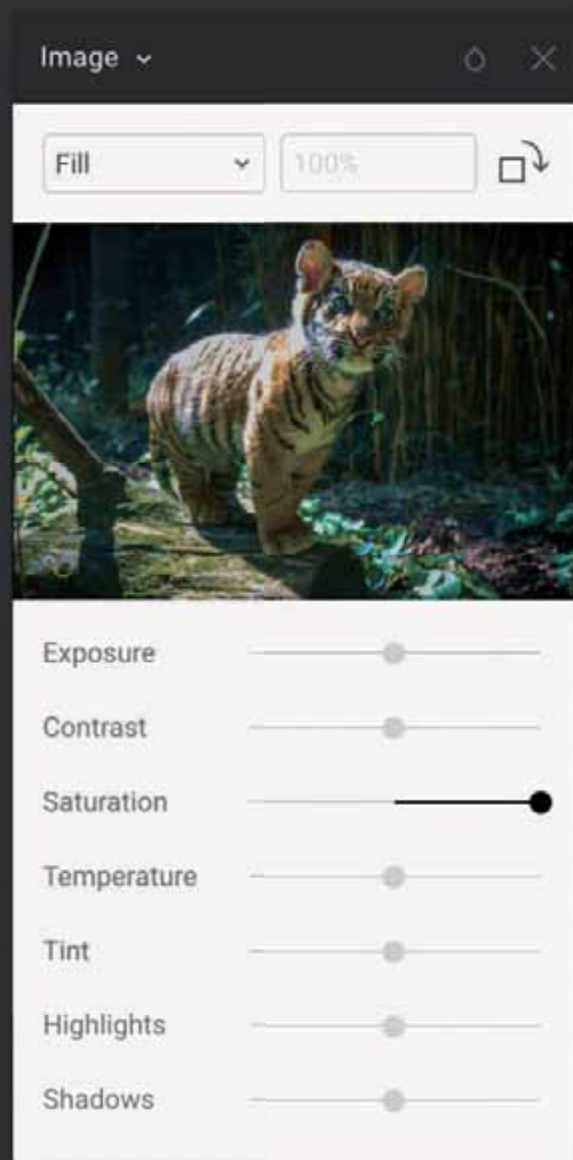
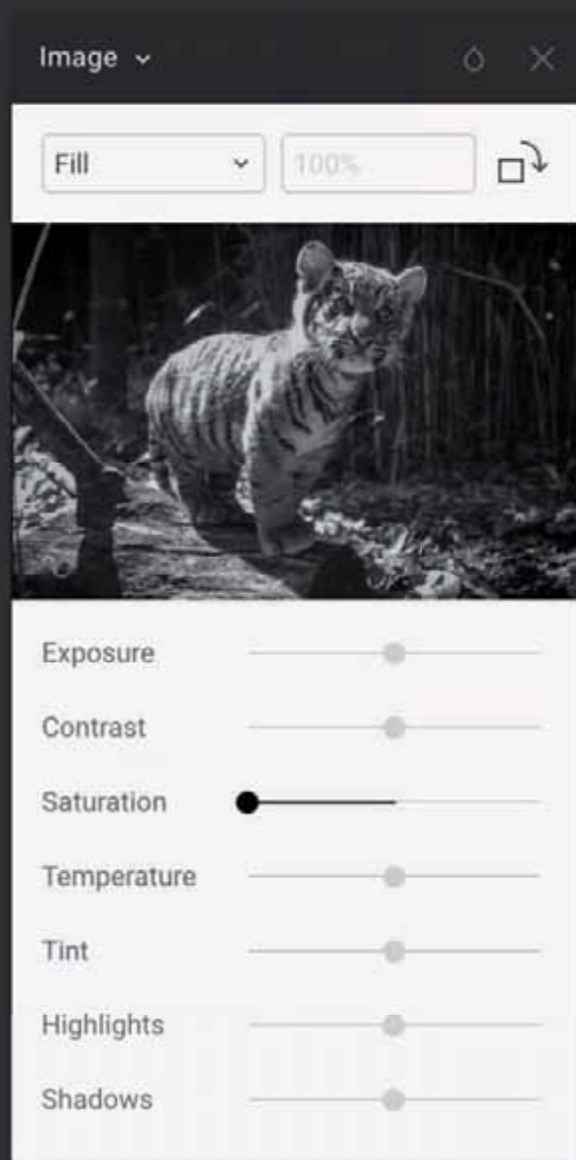
Экспозиция — базовая настройка яркости фотографии. Если кадр недодержали, он тёмный и имеет чёрные провалы. Если свет слишком долго попадал в объектив, кадр слишком светлый и имеет пересветы — аналогичные белые провалы. В фото с тигром экспозиция была оптимальной, поэтому её стоило бы оставить в нулевом значении.

Если нет цели достичь определённого художественного эффекта, и проваленных чёрных теней и пересветов следует по возможности избегать.



Contrast

Увеличение контраста позволяет делать тёмные участки темнее, а светлые светлее. Уменьшение приглушает между ними разницу. Если контраста не хватает, фотография выглядит невыразительно. Если его слишком много, переходы цвета слишком резкие. На фото справа они видны как чёрные и белые пятна.

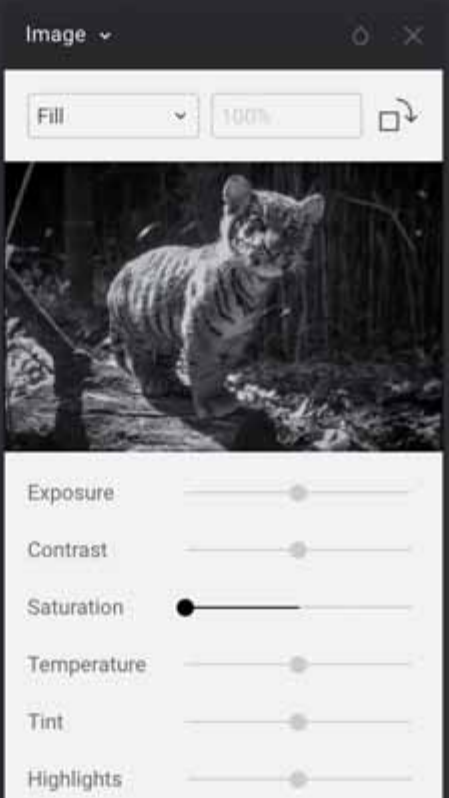


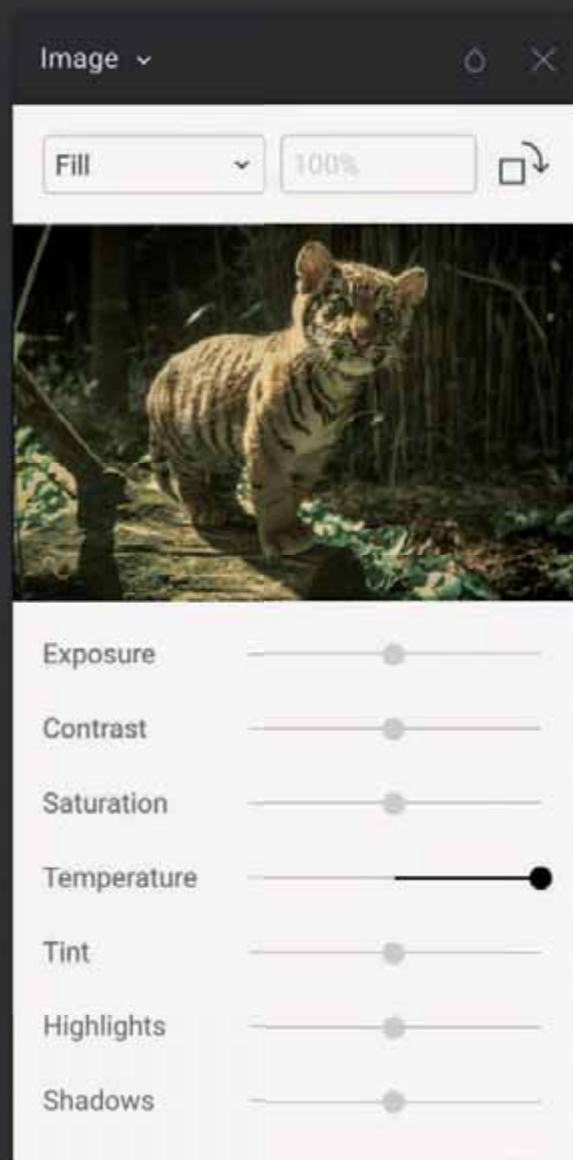
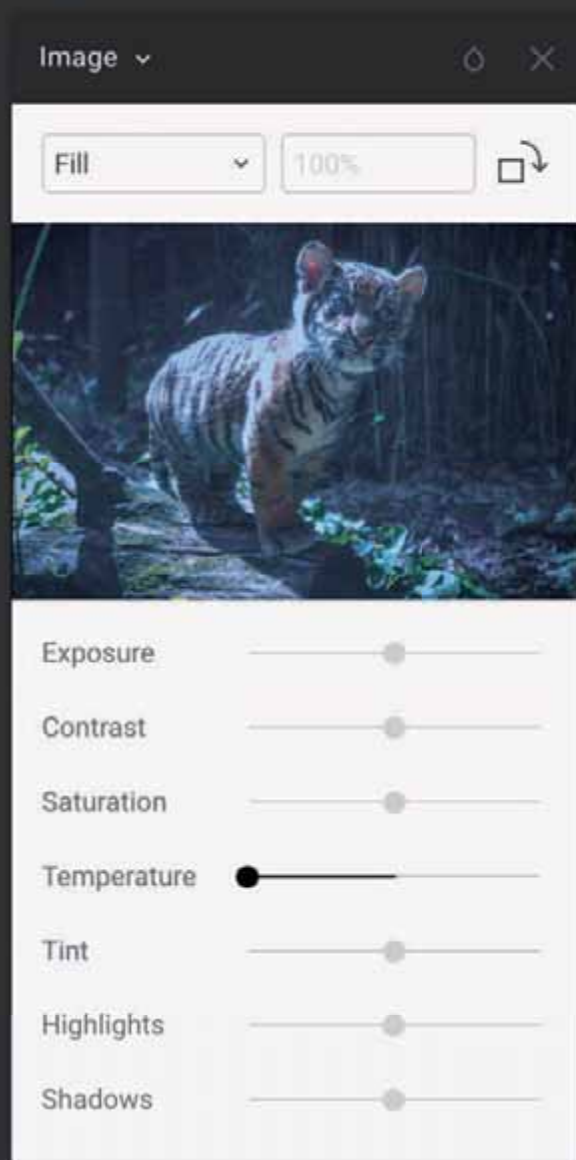
Saturation

Насыщенность оттенков, позволяет сделать цвета поярче. Если увести эту настройку в ноль, мы лишим цвета оттенков и таким образом сделаем фото чёрно-белым. Максимальное значение насыщенности делает цвета неестественно-кислотными.

Чтобы чёрно-белая фотография смотрелась сочно и драматично, увести контраст в ноль недостаточно. На фото останется много серого.

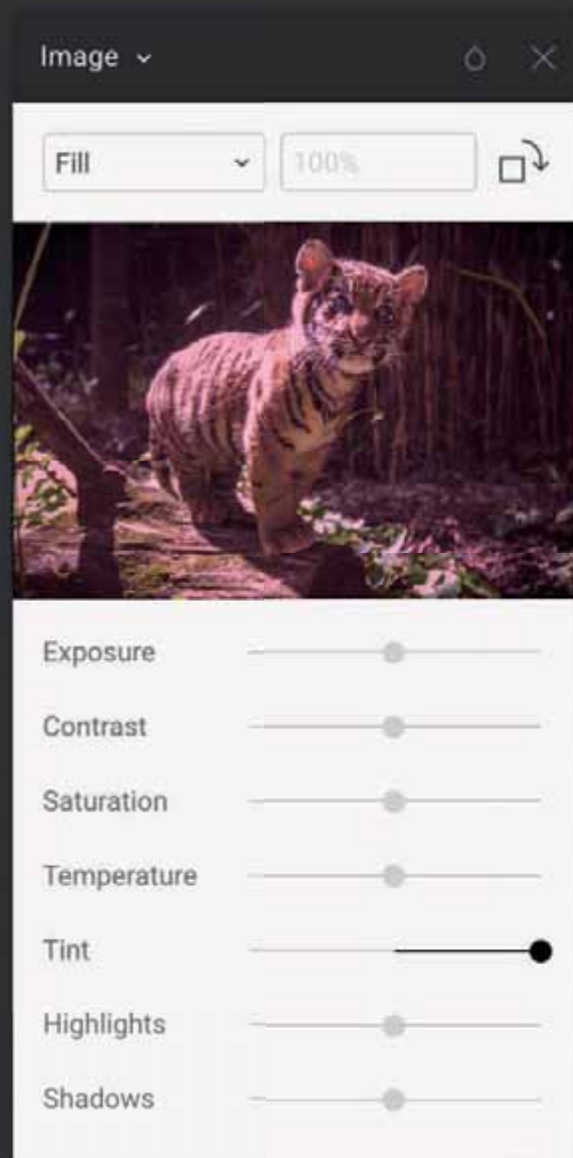
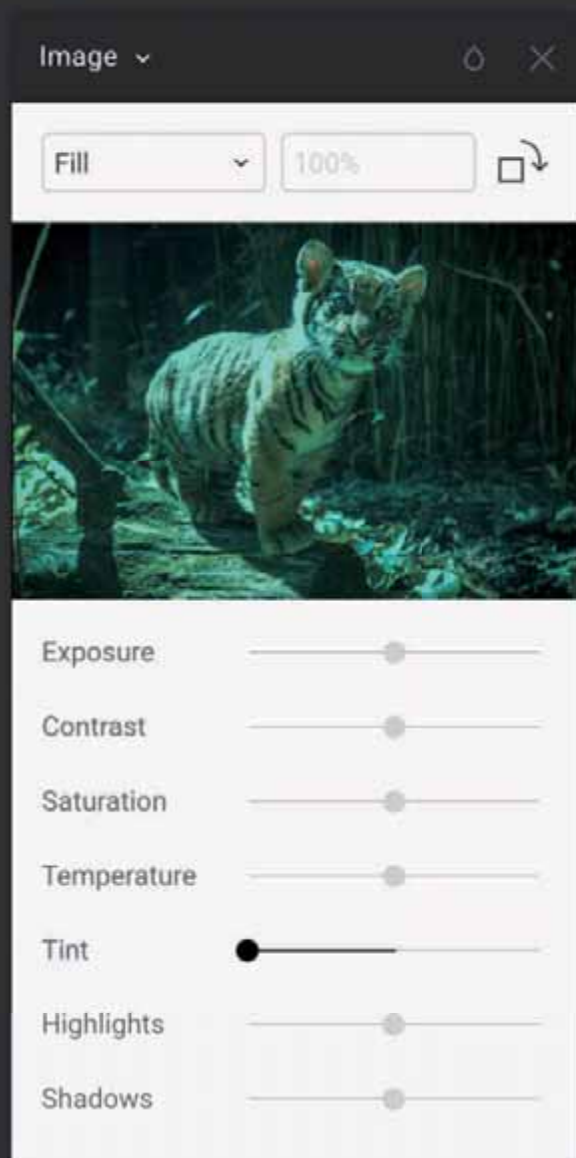
Компенсируем его высоким контрастом и глубокими чёрными:





Temperature

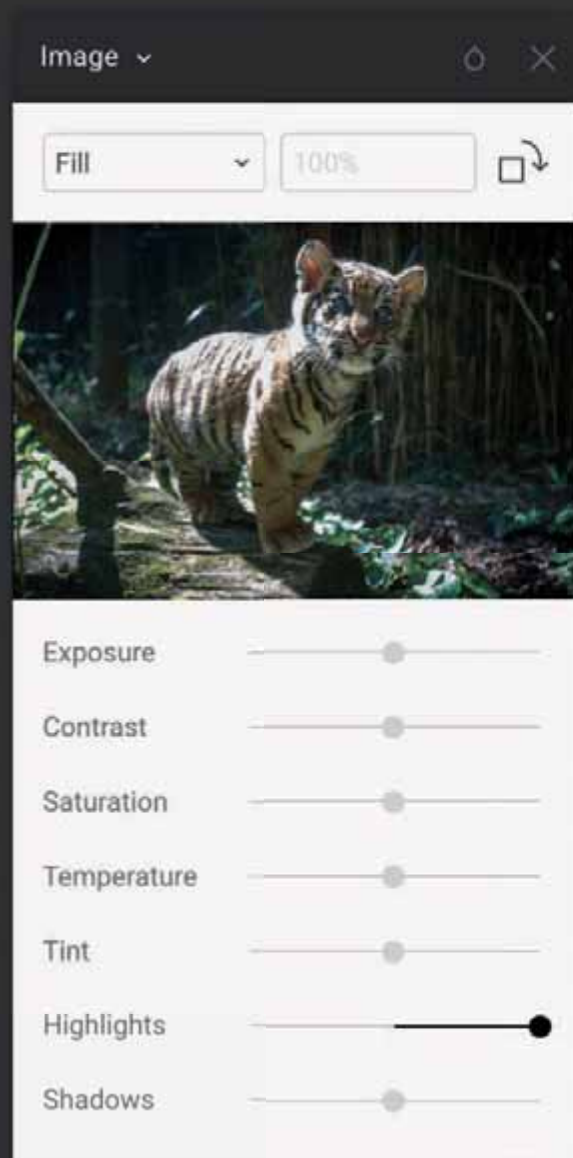
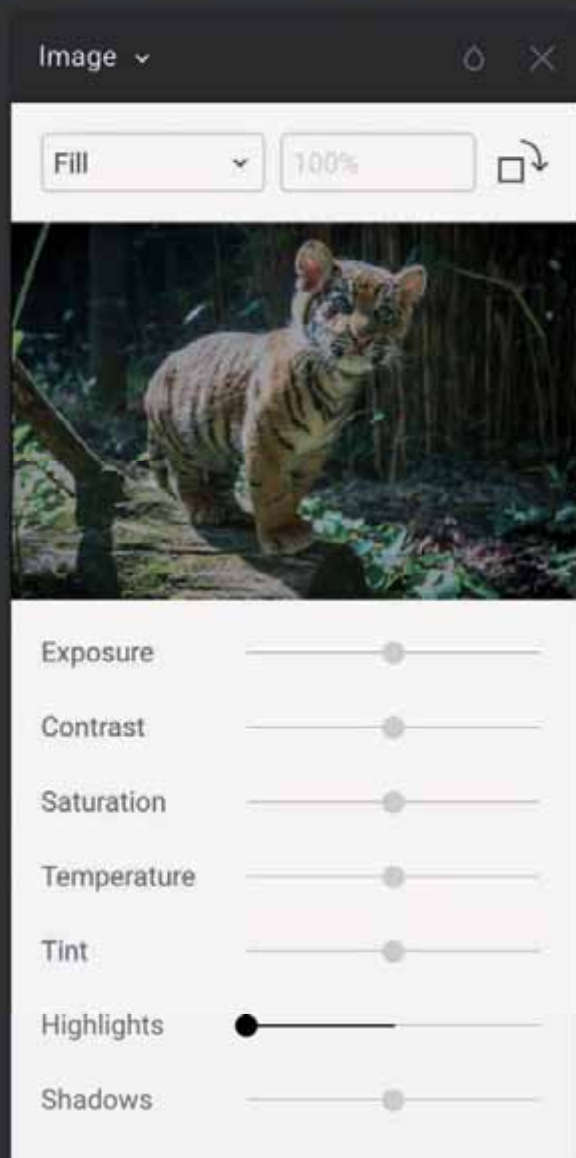
Настройка температуры света позволяет работать с балансом белого. Если кадр смотрится слишком тёплым или холодным, эта настройка позволяет корректировать его, восстанавливая нейтральное освещение. Солнечный свет днём и под вечер имеет слишком тёплый оттенок. Для восстановления нейтрального освещения он компенсируется холодным синим. Лампам люминесцентного освещения наоборот недостаёт температуры. Такие фото компенсируют жёлтым.



Tint

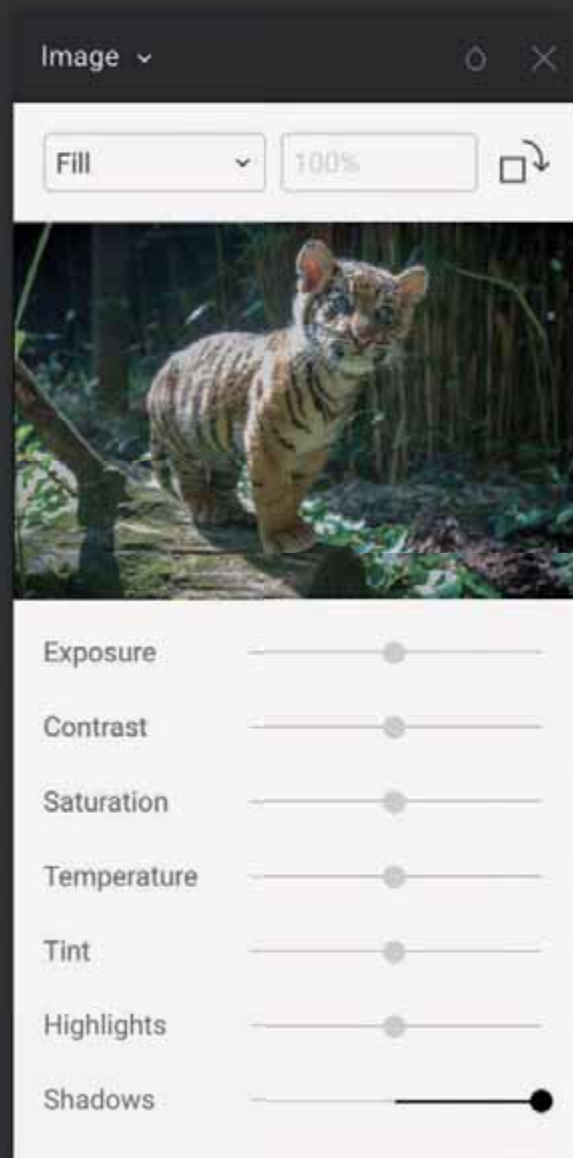
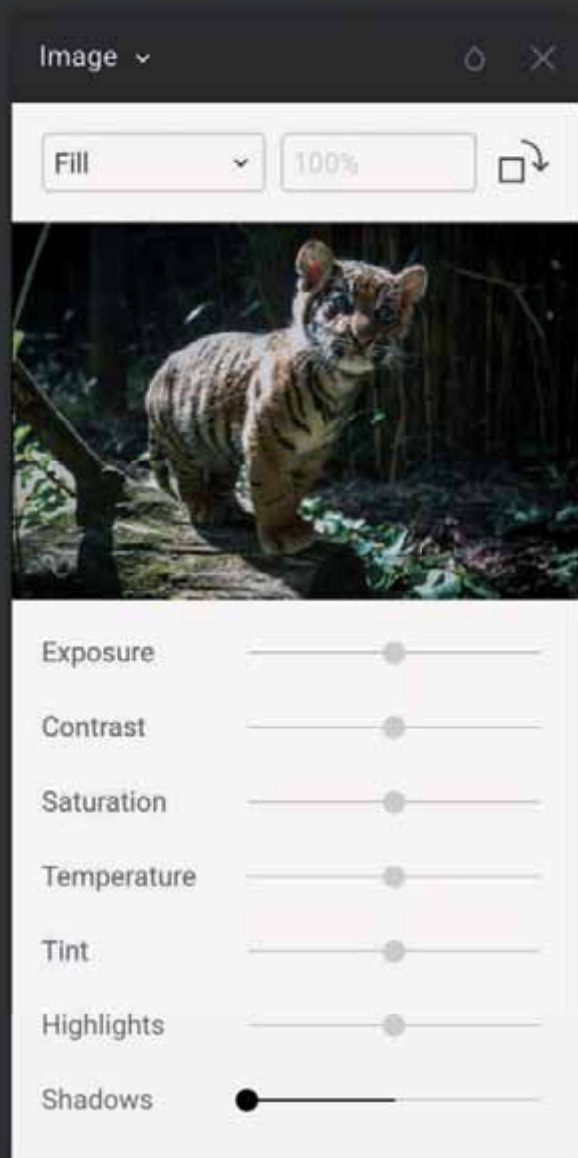
Настройка оттенка позволяет добиться эффекта выцветшего фото. Фотографы используют её, чтобы приглушать излишнюю красноту лиц, уводя оттенок немного вниз.

Полезен также для компенсации баланса белого в сложных условиях съёмки, когда не хватает обычной коррекции температуры. Например, когда на объект съёмки влияют сильно освещённые цветные стены.



Highlights

Дополнительная настройка для коррекции светлых и пересвеченных участков. Её действие похоже на контраст. Разница в том, что изменениям подвержены преимущественно яркие участки.



Shadows

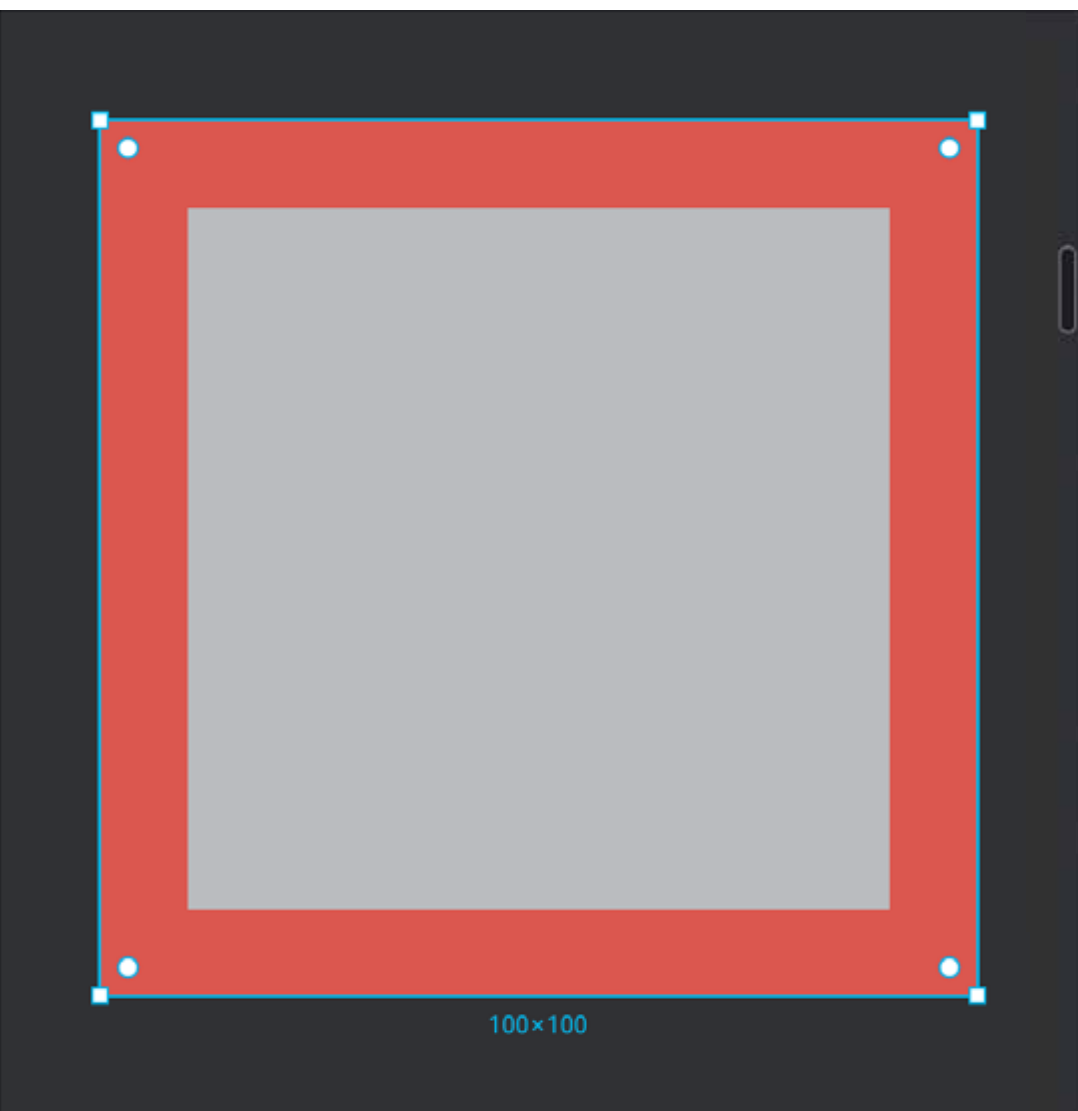
Аналог **Highlights**, только для теневых участков фото. Позволяет вытянуть излишне глубокие тени.

17. Обводка

[Проект в Фигме](#) →

В этой главе поговорим про дополнительные настройки обводки шейпов, векторных линий и текстовых слоёв, а заодно попрактикуемся их создавать. Также сравним, чем отличается реализация обводки Фигмы и Скетча.

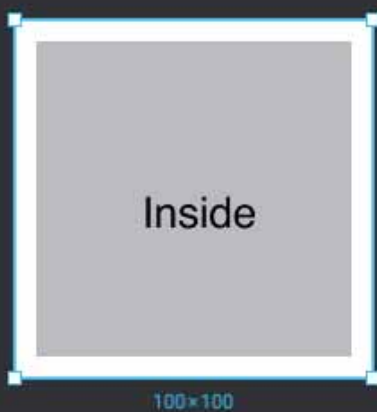
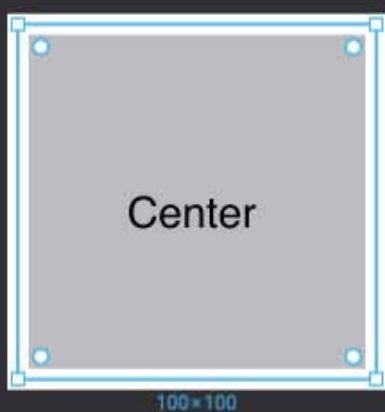
1. Создадим квадрат, **R**.
2. Зададим ему обводку красного цвета толщиной 10 пикселей.



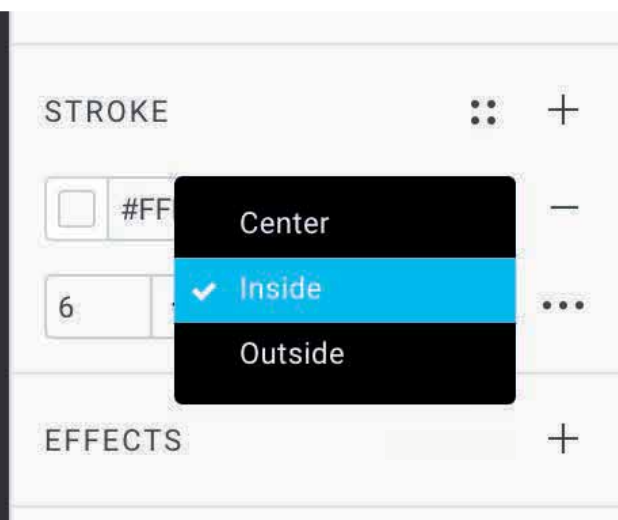
Режимы обводки

Как и в Скетче, обводка может быть наложена на фигуру тремя способами:

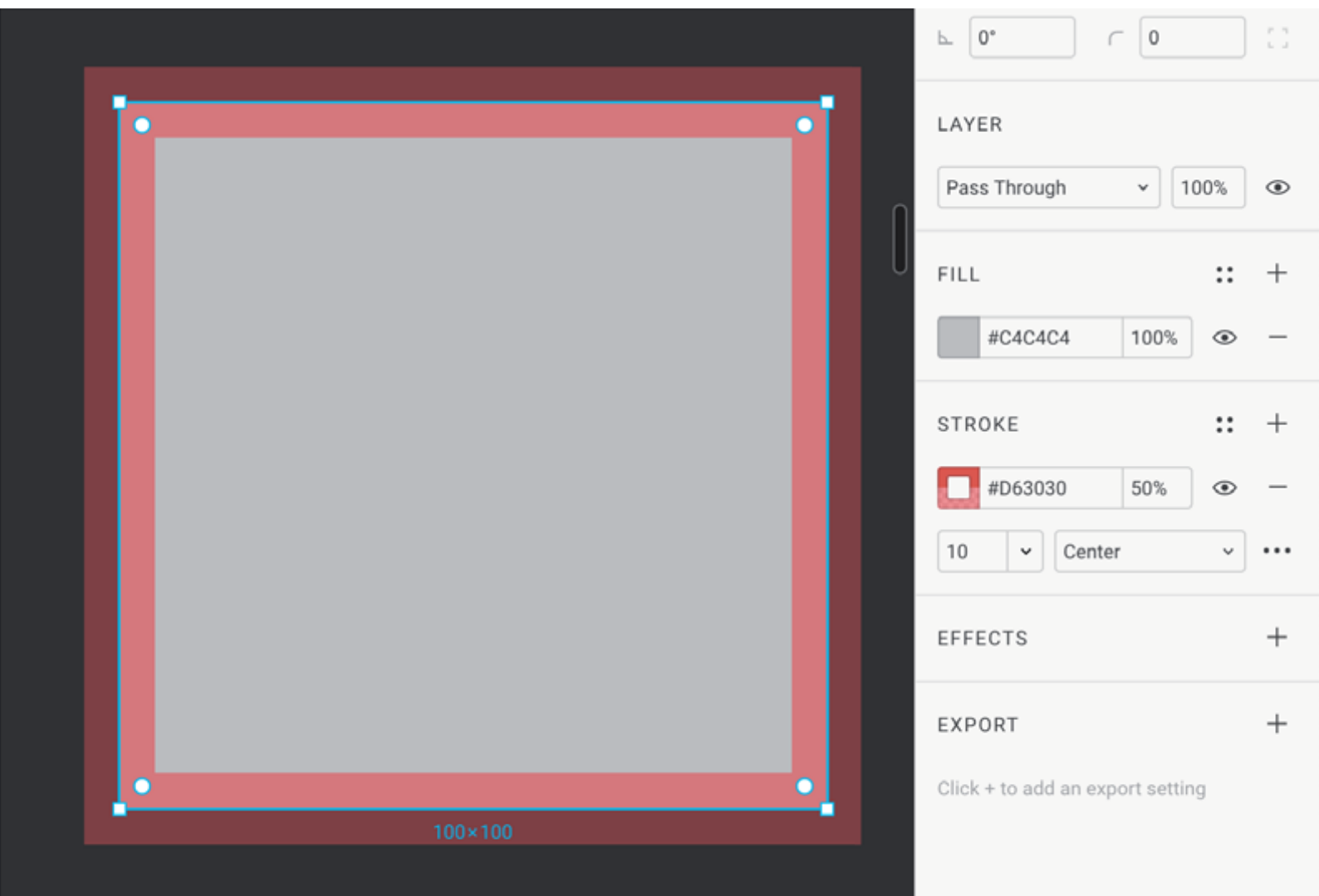
- **Center** — Центр обводки совпадает с границами шейпа.
- **Inside** — Внутри. Обводка не выходит за границы шейпа.
- **Outside** — Снаружи. Внешняя обводка как в полях ввода Google Material.



Это настраивается в выпадающем меню в блоке Stroke.



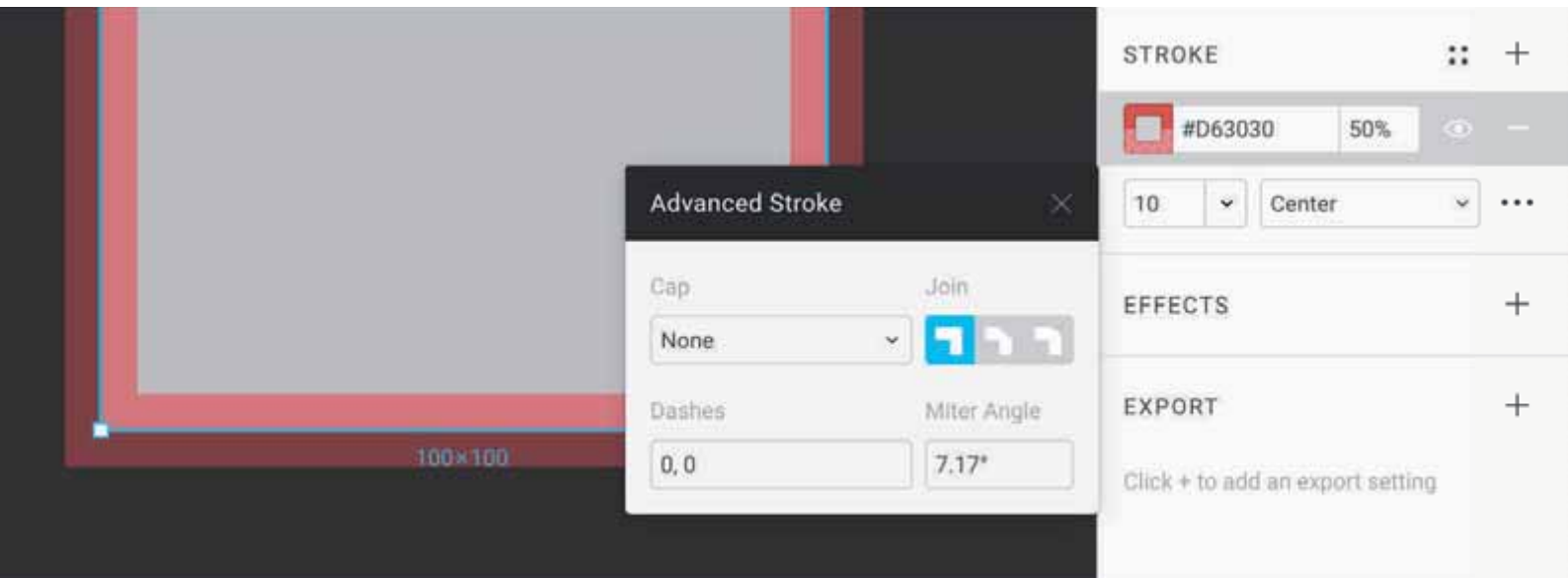
3. Поставим режим обводки с **Inside** в **Center**, а саму её сделаем на 50%. Теперь явно видно, что красная линия проходит по центру контура фигуры.



Отличия пунктирной обводки в Скетче и Фигме

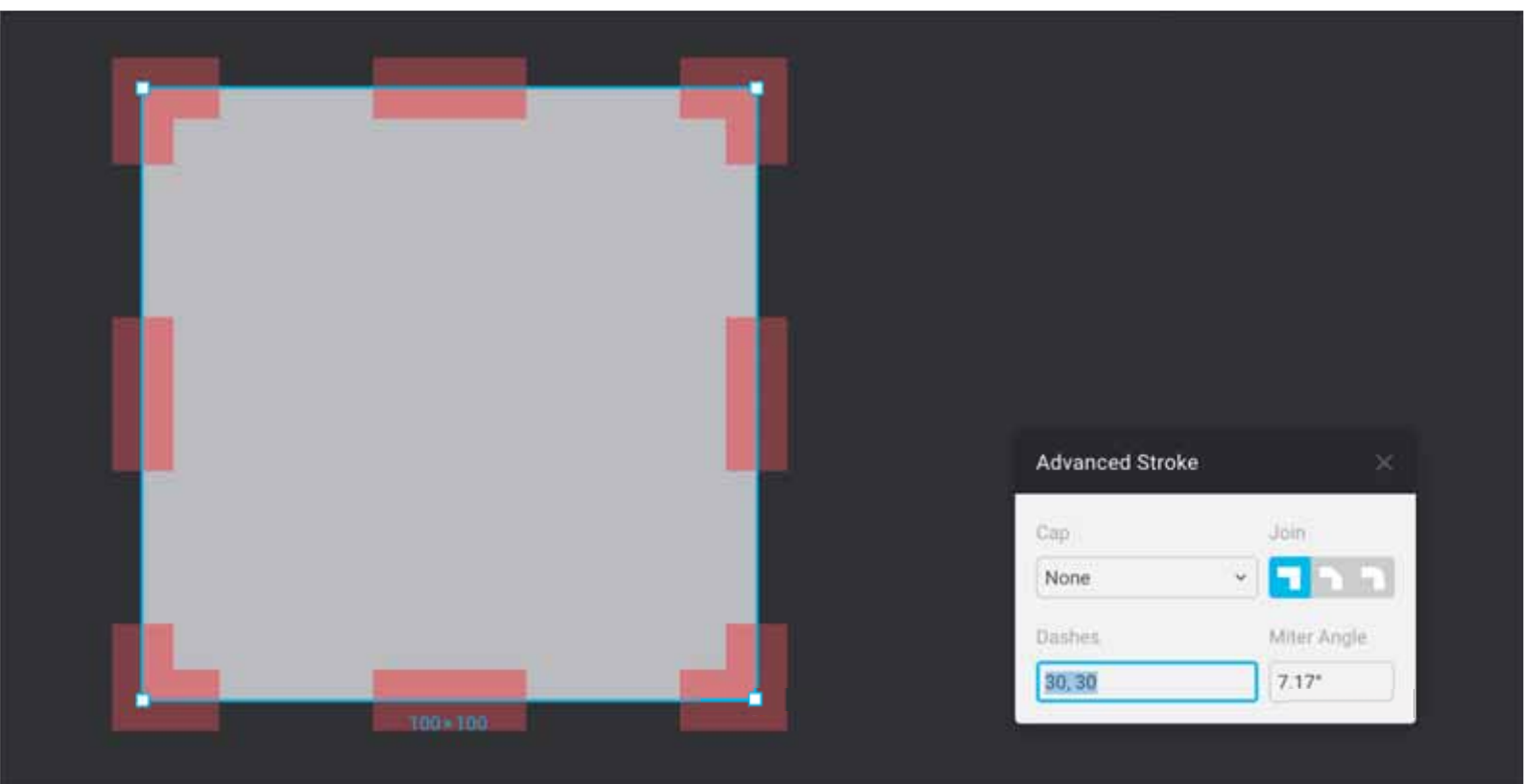
В правой части блока **Stroke** есть меню в виде трёх точек.

Разворачиваем:

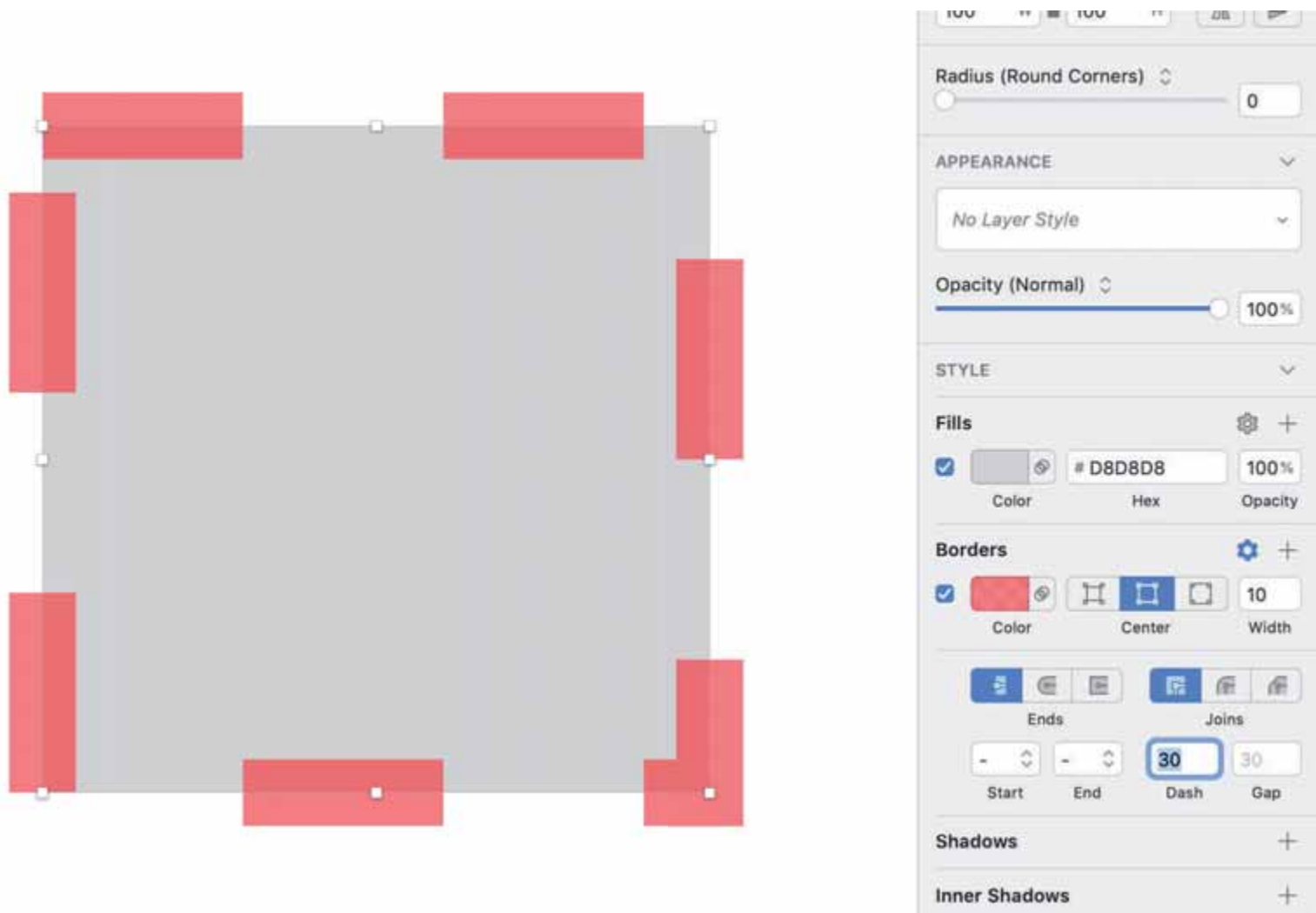


В нём в поле **Dashes** можно задать, чтобы линия стала пунктирной.

Первое значение — длина линии в пикселях, второе — отступ в пикселях до следующей. Задаём **30**.



В Скетче аналогичная фигура при таких настройках выглядит совсем иначе:



Разница в том, что Скетч механически отсчитывает линию от верхнего левого угла. Работа с таким пунктиром слабо предсказуема.

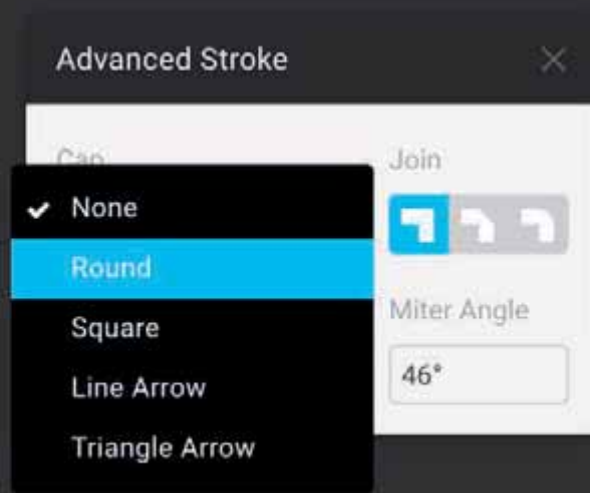
Фигма подстраивает пунктир таким образом, чтобы тот был симметричен. Красная линия обязательно попадает на углы фигуры своим центром. Так пунктир выглядит наиболее эстетично. При этом Фигма пренебрегает математически-точными значениями.

Стили изгибов обводки

У обводки в квадрате могут быть три стиля изгибов, которые переключаются в сегменте **Join**:

- **Miter** — прямой угол, по умолчанию.
- **Bevel** — срезанный на 45°.
- **Round** — закруглённый.

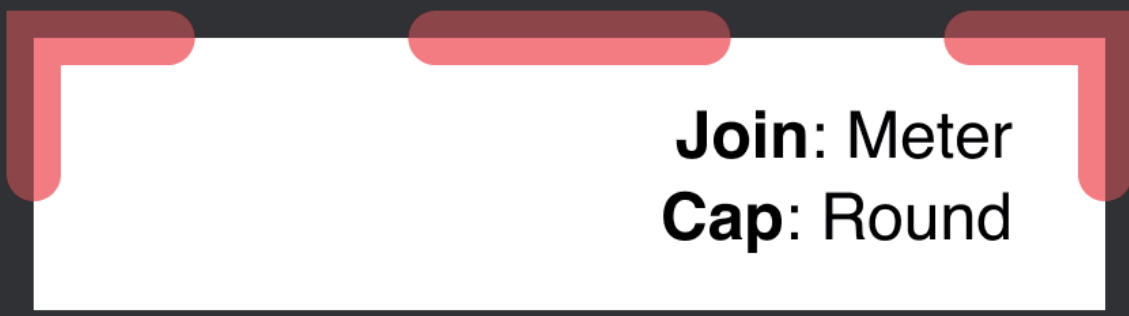




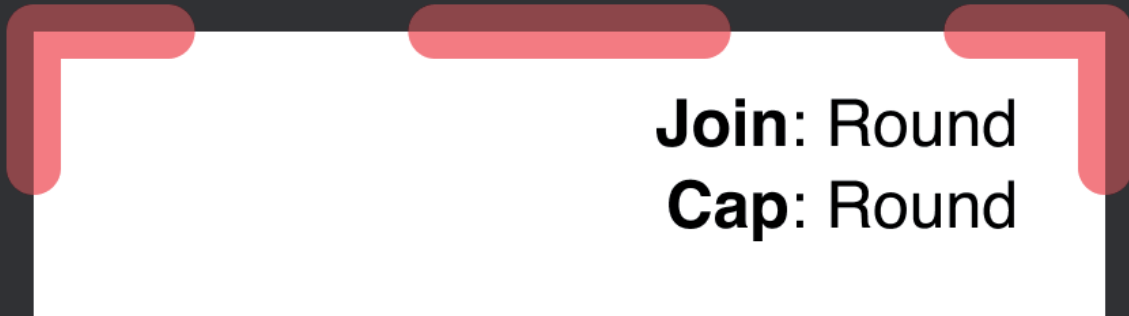
Типы окончания линии

Настройка **Cap** отвечает за то, как будет оформлено окончание линии. Мы говорили о типах окончаний, когда изучали стрелки в главе **Шейпы**.

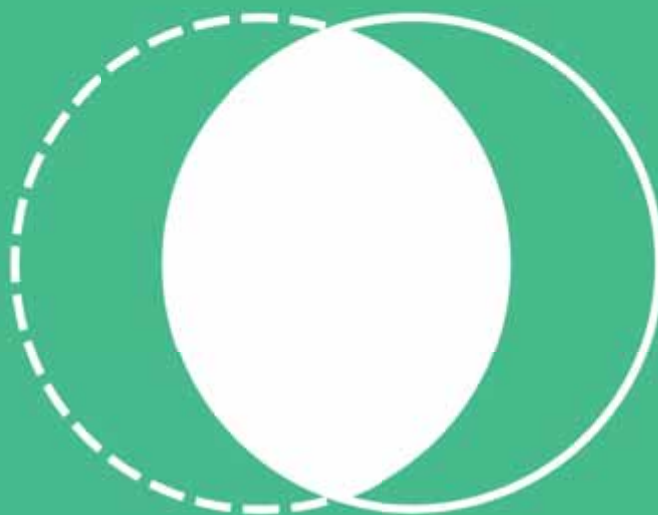
По умолчанию **Cap** установлен в **None** и прерывается. К линиям в обводке можно применить тип **Round**, чтобы скруглить пиктир.



Join: Meter
Cap: Round



Join: Round
Cap: Round



18. Маски

[Проект в Фигме →](#)

Маски позволяют брать одну векторную фигуру и по её форме делать видимой другую.

Кадрирование фото

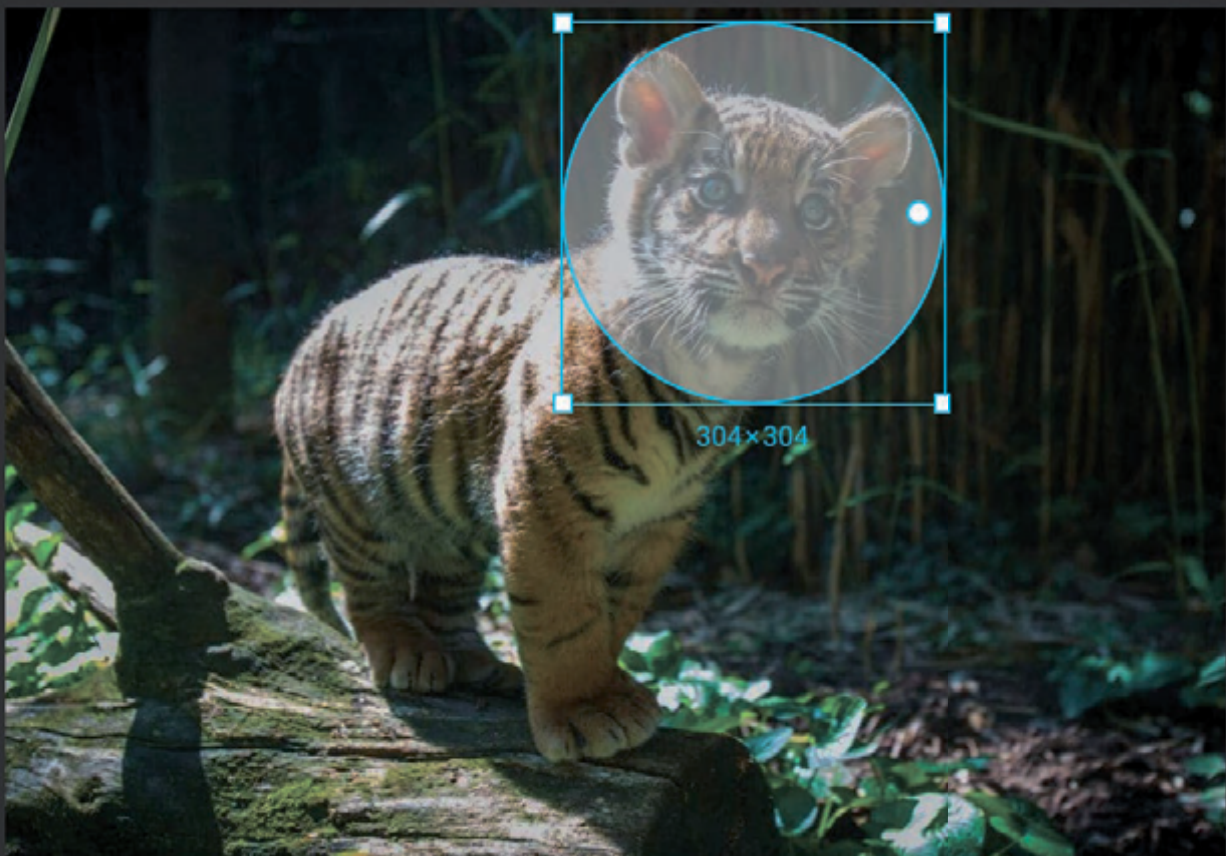
В Скетче чаще всего в качестве маски используются прямоугольник или круг, а в качестве контента — фотографии.

Чтобы добиться контроля над кадрированием без использования масок, нужно заранее подготовить все изображения и вставлять их в заливку в виде уже обрезанных битмапов. Это костыльный и негибкий вариант.

Поэтому, маски стали у меня основным способом для создания круглых аватаров и прочих иллюстраций в Скетче.

Поскольку в Фигме есть очень развитый режим кадрирования заливки **Crop**, нет смысла использовать маски для кадрирования аватаров и иллюстраций. Однако, для понимания принципа их работы полезно создать простейшую маску. Механика полностью аналогична Скетчу.

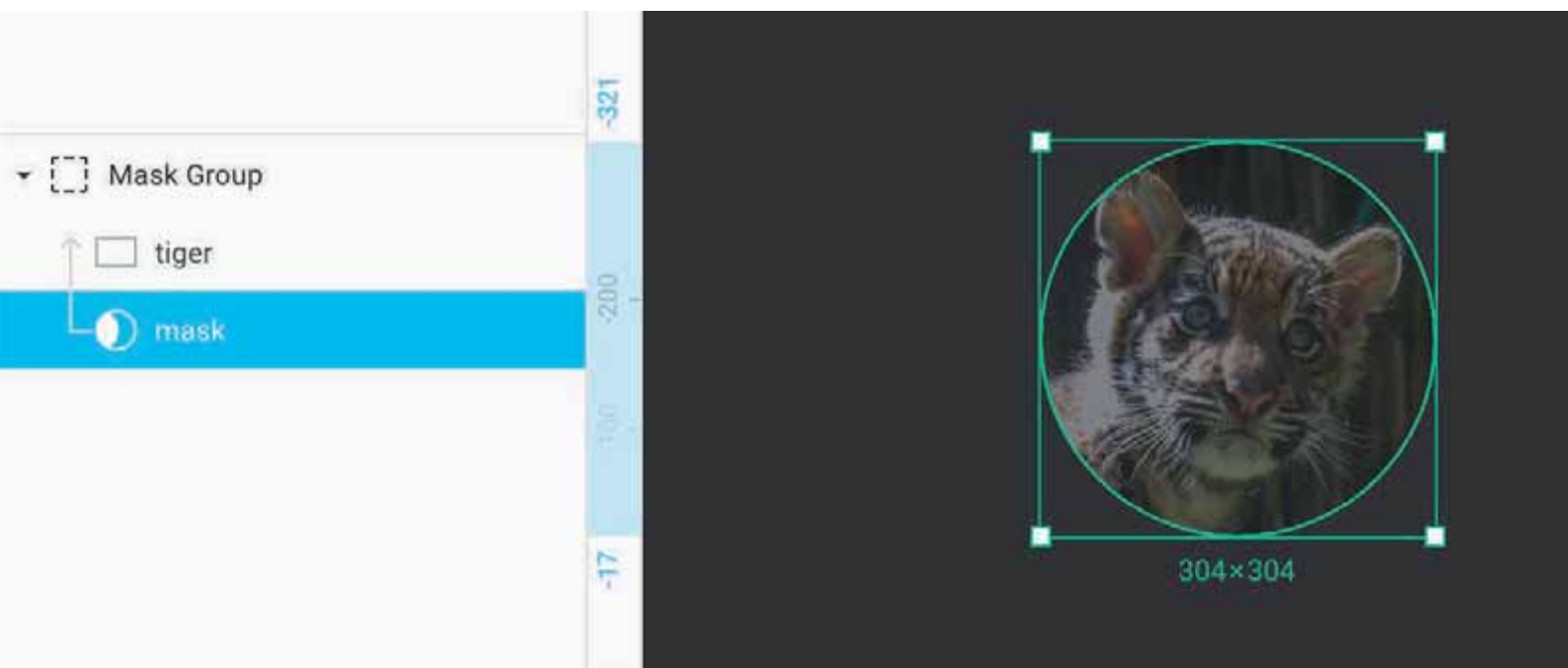
1. Вставим в рабочую область фотографию.
2. Поверх неё растянем круг, **O**. Он должен загоразивать то место в фотографии, которое мы собираемся оставить видимым.
Для наглядности можно понизить опасность круга до 50%, **5**.
3. Круг выделен. Поменяем его местами с фотографией, **Cmd + [**. Круг должен встать снизу.
4. Зажимаем **Shift** и выделяем оба слоя.



5. Вызываем команду **Use as Mask**, **Ctrl + Cmd + M** или кликаем тулбар:



Была создана новая группа с названием **Mask Group**. Слой с кругом теперь находится в режиме маски и поэтому его бокс и контур кодируются зелёным цветом.

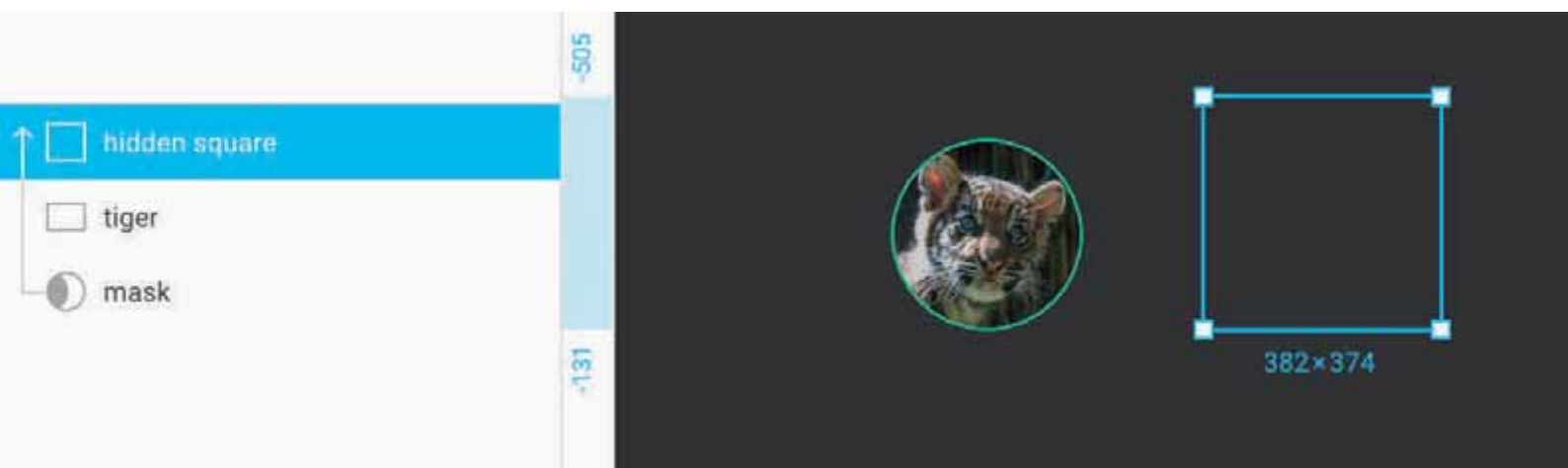


Пока слой фотографии находится над маской, он будет обрезаться по её форме.

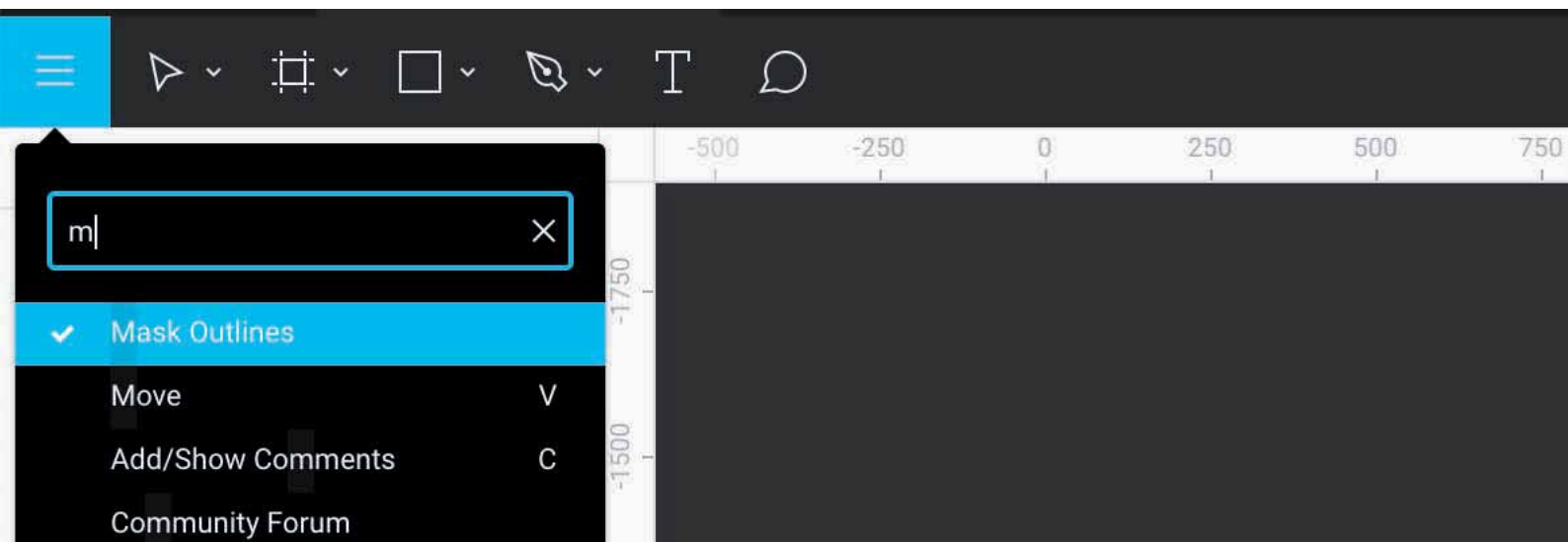
Поскольку у круга была задана опасность 50%, фотография тоже её наследует. Возвращаем опасность в максимум, **100**.

Слои выстраиваются в иерархию при помощи групп. Если групп нет, все слои расположены на высшем уровне.

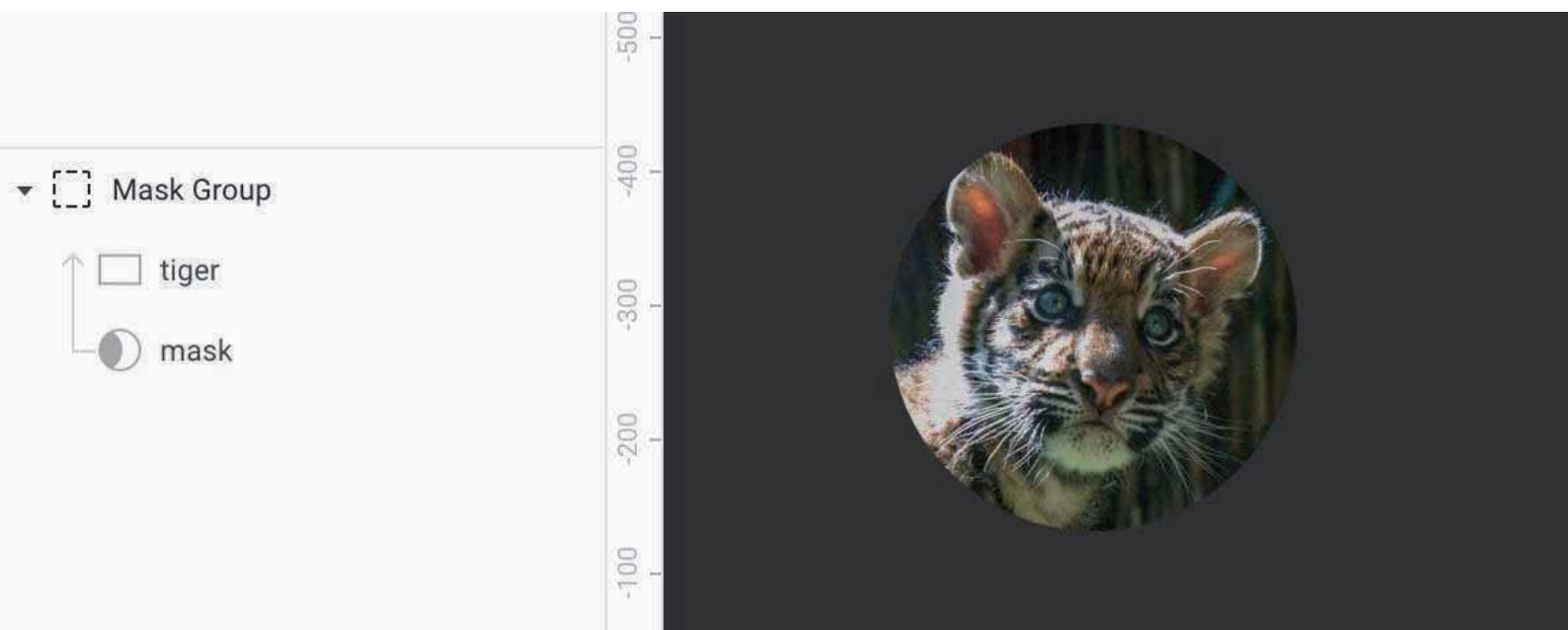
Маска действует только в рамках своей группы, поэтому, чтобы избежать попадания в неё посторонних слоёв, маску и её контент инкапсулируют в группу. Если её развязать, **Shift + Cmd + G**, маска сожрёт все слои выше себя, доступные на своём уровне иерархии.



У Фигмы есть тогл-режим **Mask Outlines**, который показывает границы масок зелёной обводкой. Его можно отключить клавишей **Cmd + /, m** или через меню.



Зелёная граница скрылась:



Альфа-маска

Такой приём используется в дизайне, если надо показать, что справа или слева ещё есть контент и его можно тянуть вправо-влево.

iPhone

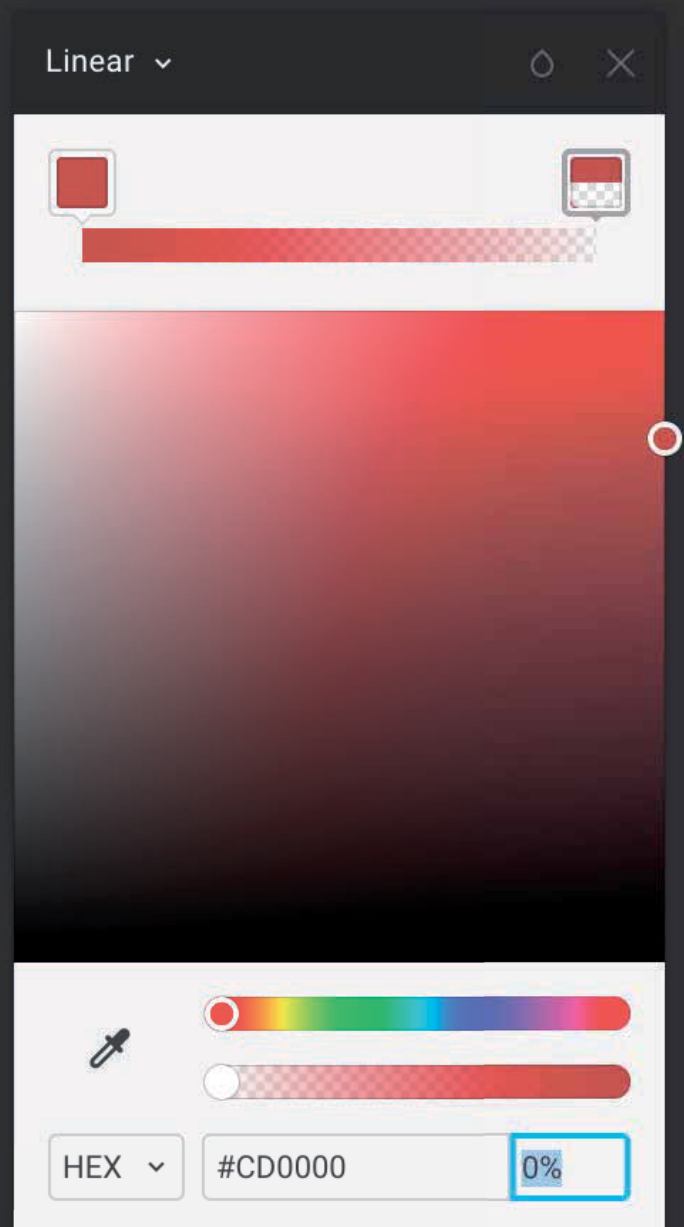
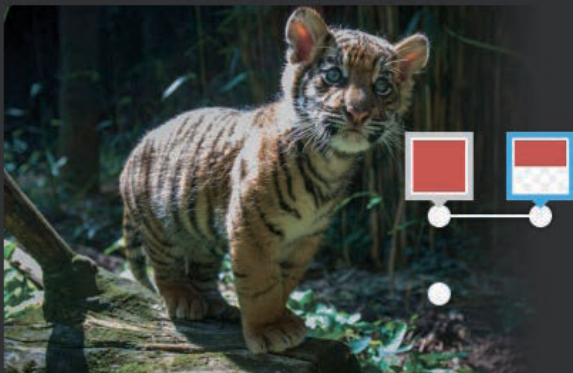


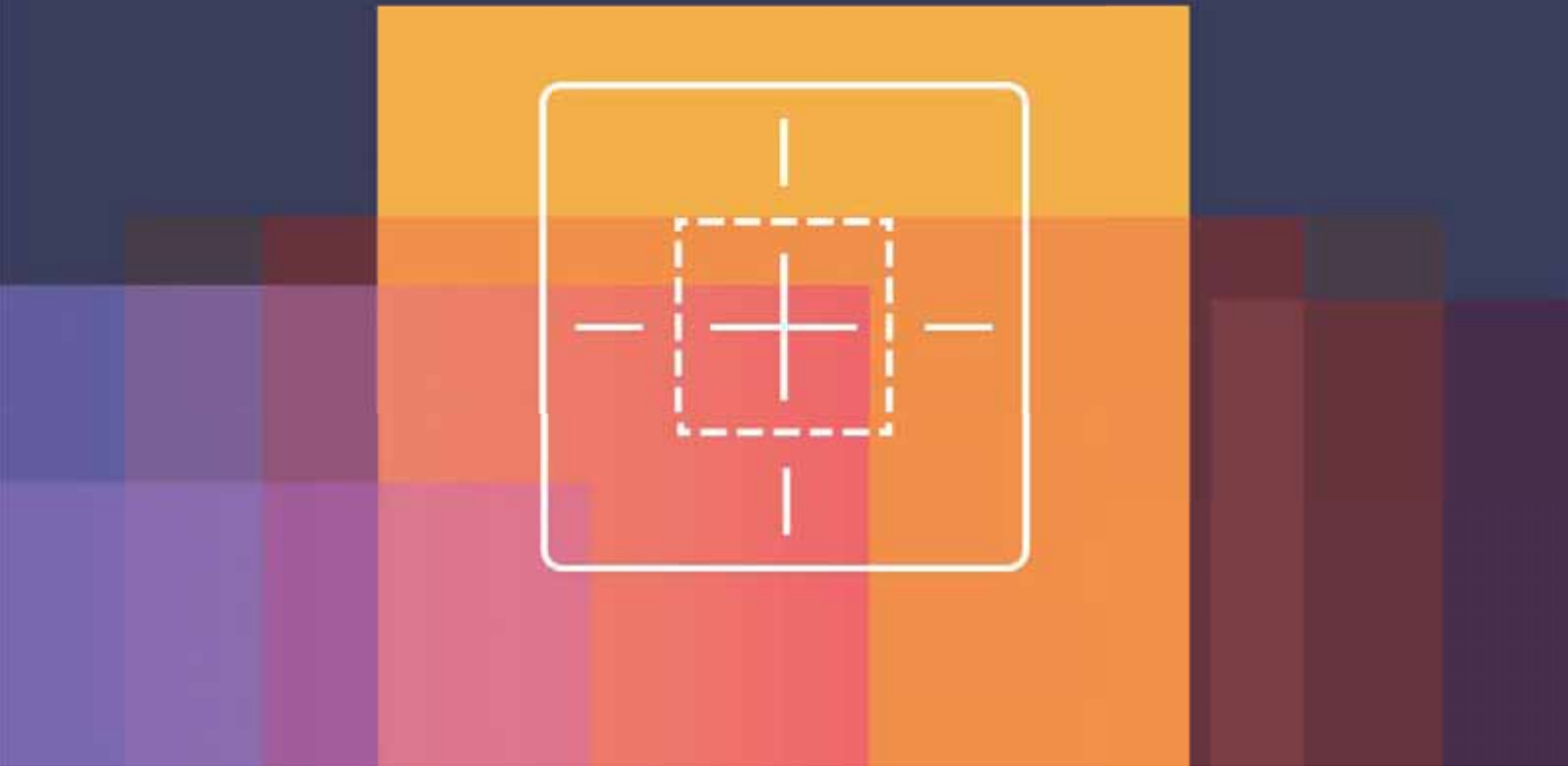
В Скетче маска может быть в двух режимах: в базовом **Outline Mask** и в **Alpha Mask**. Второй режим позволяет настраивать плавные переходы прозрачности контента. Опасность слоя-маски ни на что не влияет.

В Фигме такого излишнего деления нет, поскольку контент маски чувствителен к её опасности. Если в маске есть градиент или сама она прозрачна, такая маска будет уводить в прозрачность свой контент без дополнительного переключения режим альфа-маски.

Задача: сделаем альфа-маску.

1. Сделаем обычную маску по инструкции выше.
2. Выделим слой маски.
3. Наложим на него линейный градиент. Цвет не важен, потому что если шейп с градиентом используется как маска, Фигма будет использовать его градиент для работы с прозрачностью контента. Для наглядности выбрал красный.
4. Конечную контрольную точку градиента уводим в ноль.
5. Видим, как правая часть фотографии плавно исчезает.





19. Адаптивность и ограничители

[Проект в Фигме →](#)

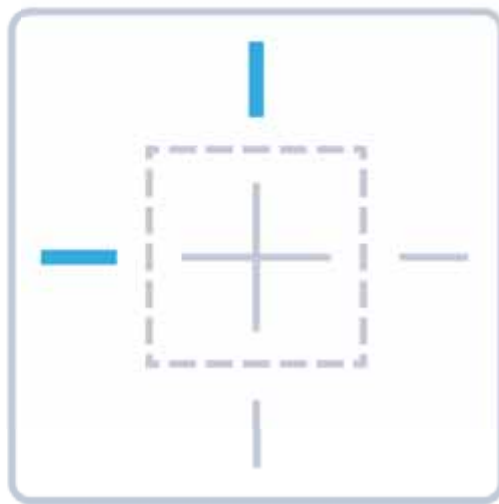
Дизайн для экранов редко проектируют под один размер. Чтобы облегчить создание макетов в меняющемся размере фрейма, в Скетче и Фигме используются ограничители.

Это важная тема, которая стоит в основе работы с компонентами и типографикой.

Ограничители — это индивидуальные настройки каждого слоя в фрейме, которые помогают контролировать, как он будет вести себя при изменении размеров фрейма.

Мы задаём поведение слоёв, чтобы использовать их в разных макетах. Можно настроить шапку сайта или приложения один раз, а затем переиспользовать её компонент на макетах любой ширины от самого маленького телефона с шириной 320 px до гигантского моноблока шириной 1920 px. Каждый слой будет иметь набор настроек, который определяет, будет ли он искажаться, липнуть к краям или вставать по центру.

Если бы не существовало ограничителей, пришлось делать шапки под каждый размер экрана, а это неэффективно.



Ограничители шейпа по умолчанию

Когда мы создаём слой в фрейме или перемещаем существующий слой в фрейм, у того появляются два ограничителя: **Left** и **Top**. Они видны в *прицеле* в блоке **Constraints**.

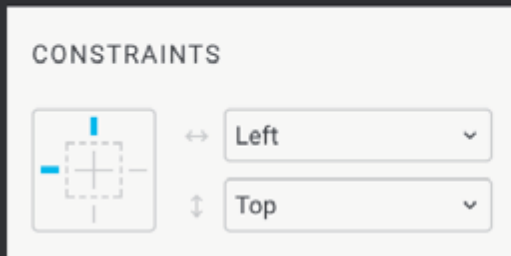
1. Создадим фрейм по размеру iPhone SE, **F**. Самый маленький айфон, который имеет смысл поддерживать. Он будет размера 320 x 568.
2. Настроим ему сетку 8px в блоке **Layout Grid**. Обычно я использую 4px, но для примера удобнее взять сетку покрупнее.

3. Растянем прямоугольник от верхнего левого края, сделав его на отступе 8 с трёх сторон. Его координаты должны быть (8, 8), а размер 304 x 56.

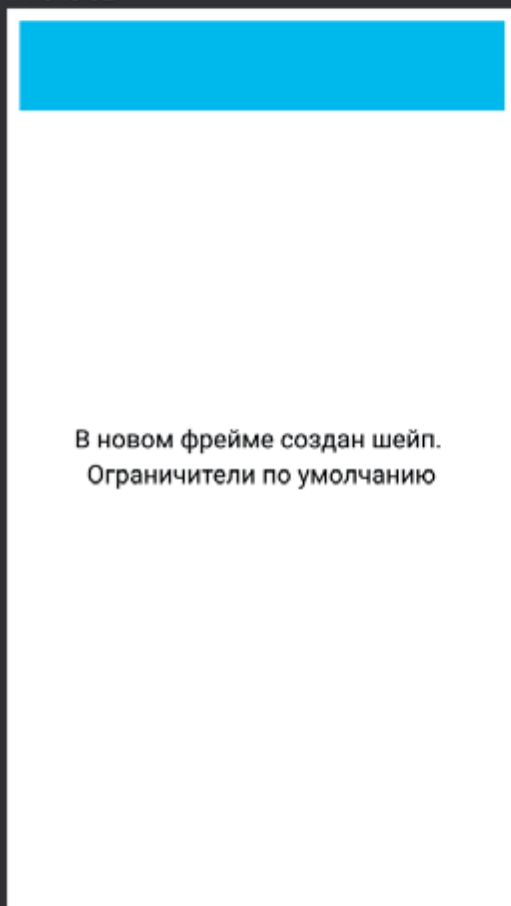


При изменении размера фрейма прямоугольник с такими ограничителями будет сохранять расстояние от края слева и сверху, а не будет растягиваться, съезжая с сетки. Размер фрейма может меняться, но отступ от края и размер фигуры останутся фиксированными. Докажем это на практике.

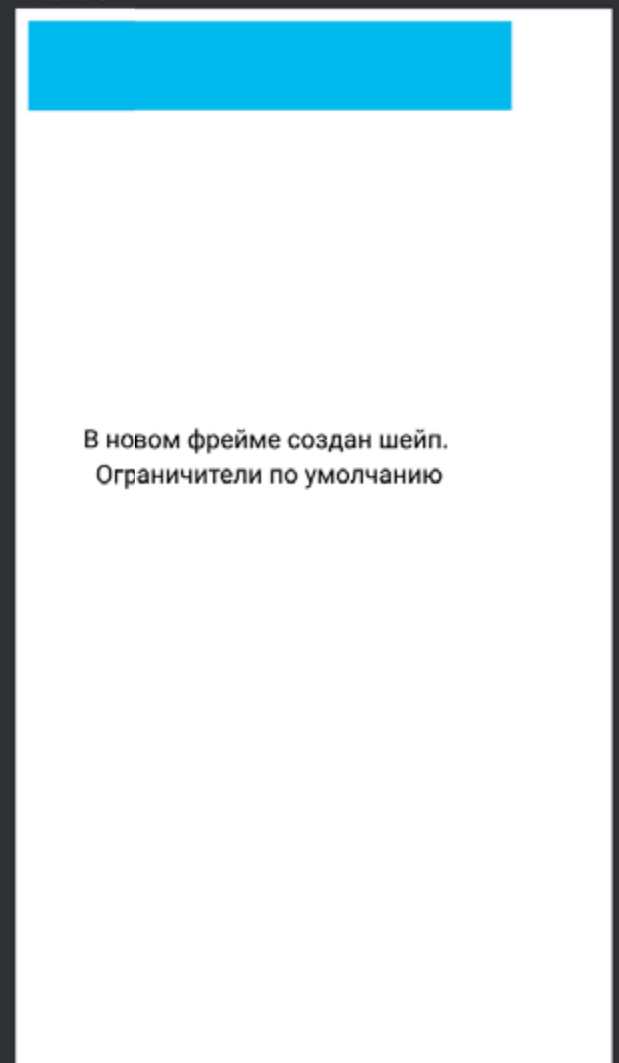
4. Выделим фрейм и зададим вместо iPhone SE другой размер: iPhone 8. Ширина увеличится до 375. Положение и ширина прямоугольника останутся прежними.

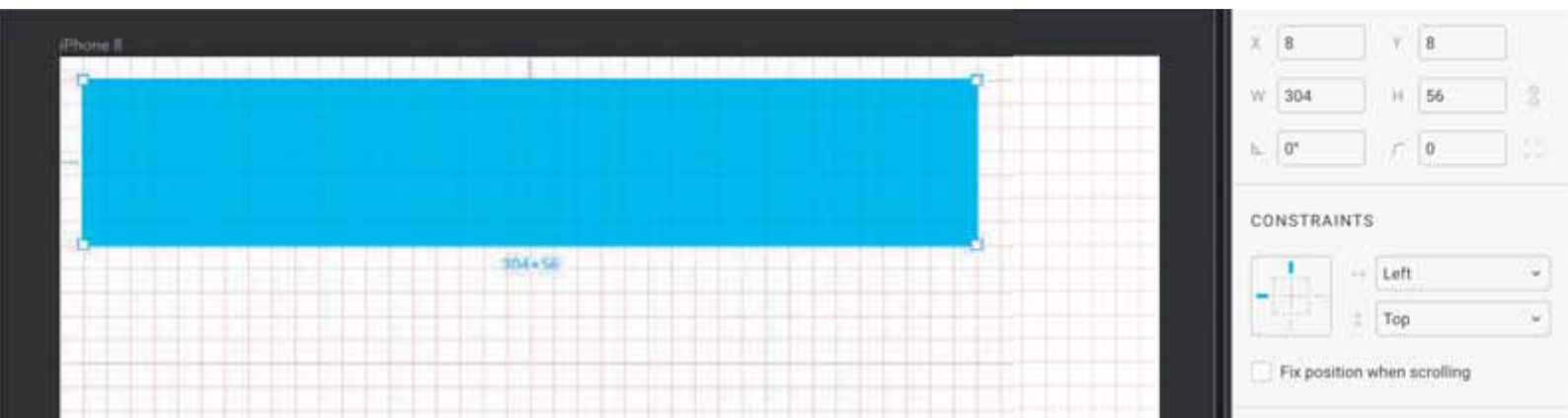


iPhone SE



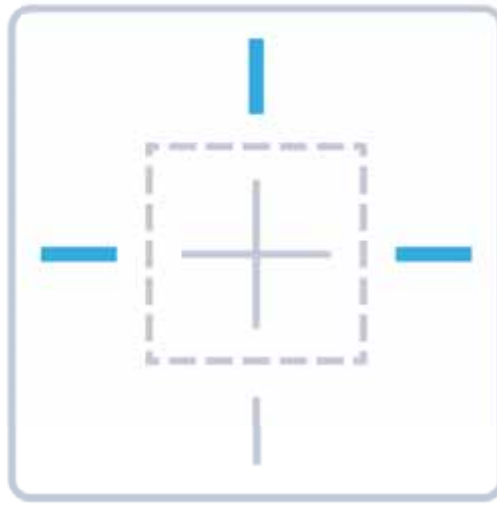
iPhone 8





В Скетче такая настройка соответствует отсутствию ограничителей и снятой галочке **Adjust content on resize** в настройках артборда.





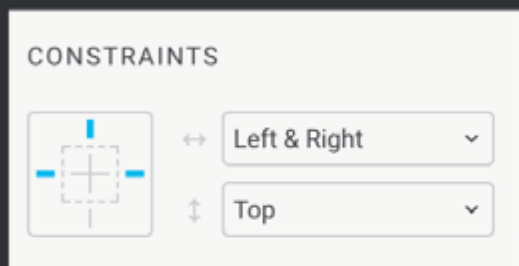
Как сделать резиновую шапку

Задача: при изменении ширины блока шапка должна растягиваться по длине, сохраняя отступы от края фрейма и свою высоту.

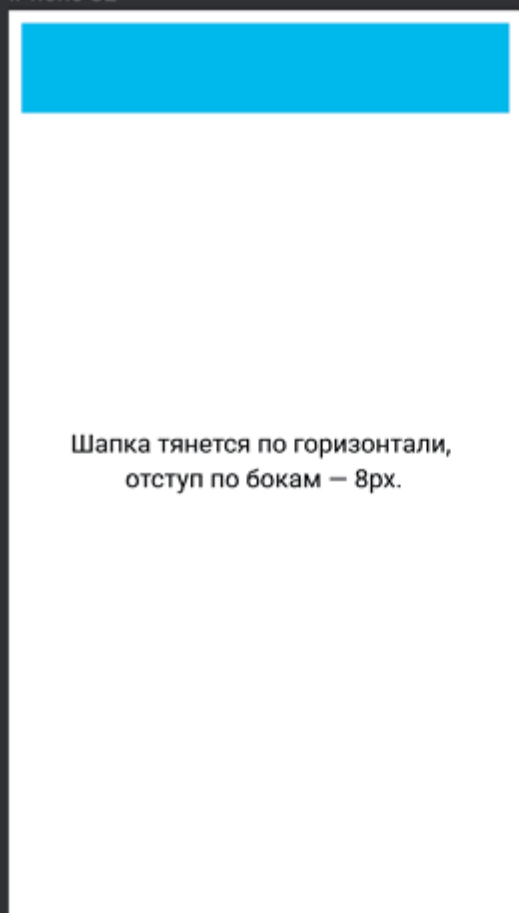
1. Возвращаем размер фрейма на iPhone SE.
2. Выделяем прямоугольник, зажимаем **Shift** и кликнем в правую чётточку, чтобы закрепить ограничители слева и справа. Также это можно сделать через выпадающее меню.
3. Теперь горят три из них, а вместо **Left** в меню горизонтальных ограничителей стало **Left & Right**:



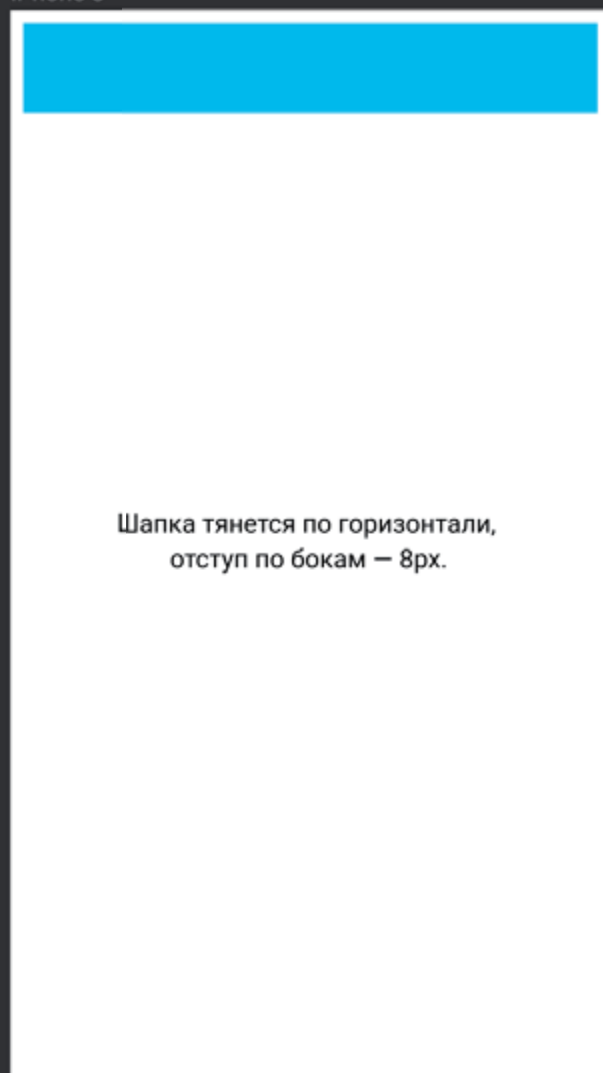
4. Снова ставим размер фрейма в iPhone 8, чтобы проверить, как поведёт себя блок. Теперь правый отступ стал фиксированным. Он был равен 8 при ширине фрейма 320 и остался равен 8 при ширине 375.



iPhone SE



iPhone 8

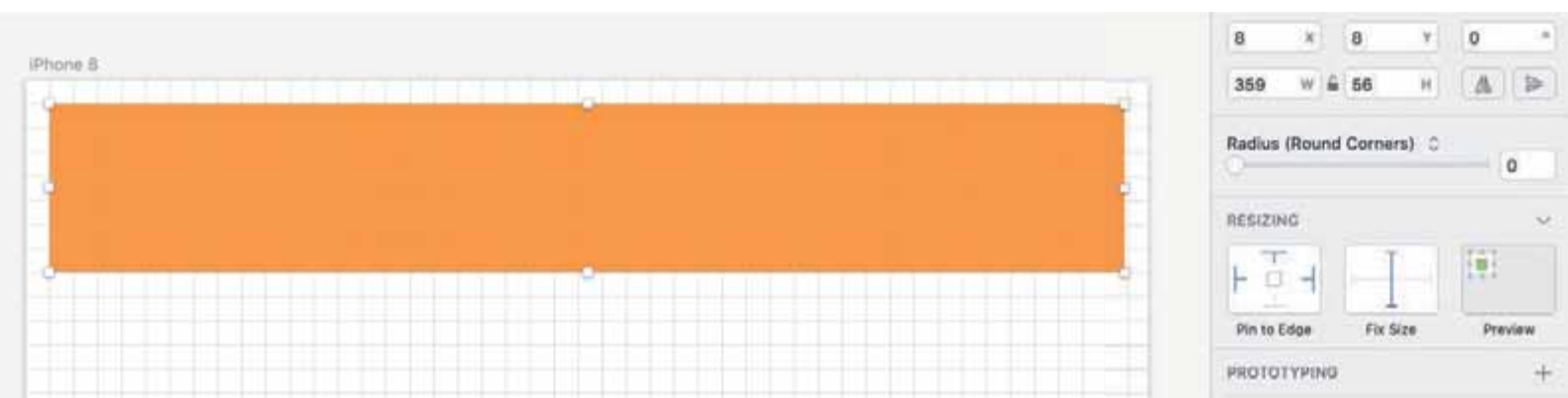


Как мы и хотели, ширина прямоугольника увеличилась: была 304, а стала 359. Высота сохранилась в 56.



В Скетче этой настройке соответствует такая комбинация ограничителей:

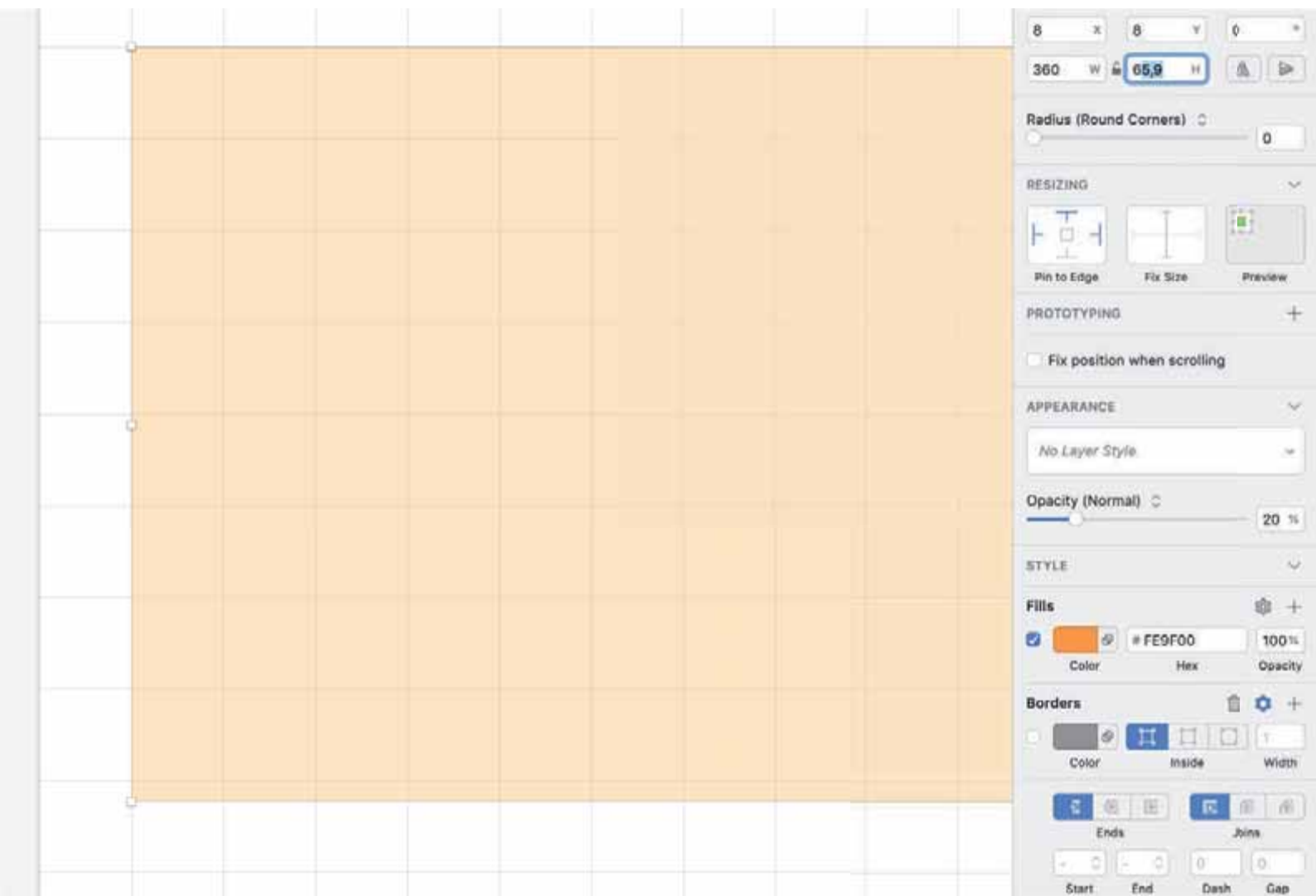
- Pin to Left
- Pin to Top
- Pin to Right
- Fix Height
- В настройках артборда стоит галочка **Adjust content on resize**



Галочка **Adjust content on resize** в Скетче

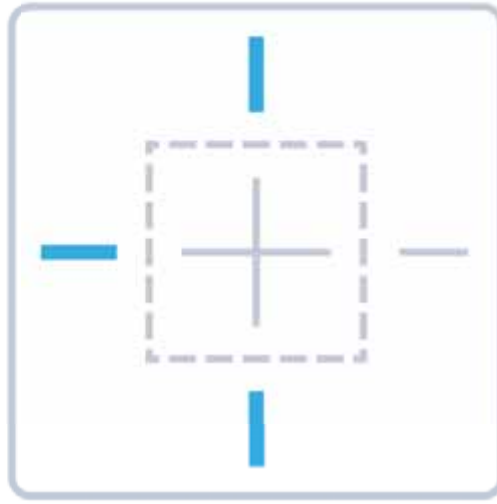
В выключенном состоянии она позволяет игнорировать ограничители и растягивать артборд.

Если оставить её в настройках артборда, при переключении с iPhone SE на iPhone 8 прямоугольник увеличится пропорционально ширине артборда и съедет с сетки.



Чтобы этого не произошло, перед растягиванием артборда нужно дополнительно зафиксировать его высоту ограничителем **Fix Height**. Аналогичный ограничитель бывает для ширины.

В Фигме же ширина и высота изначально зафиксированы.



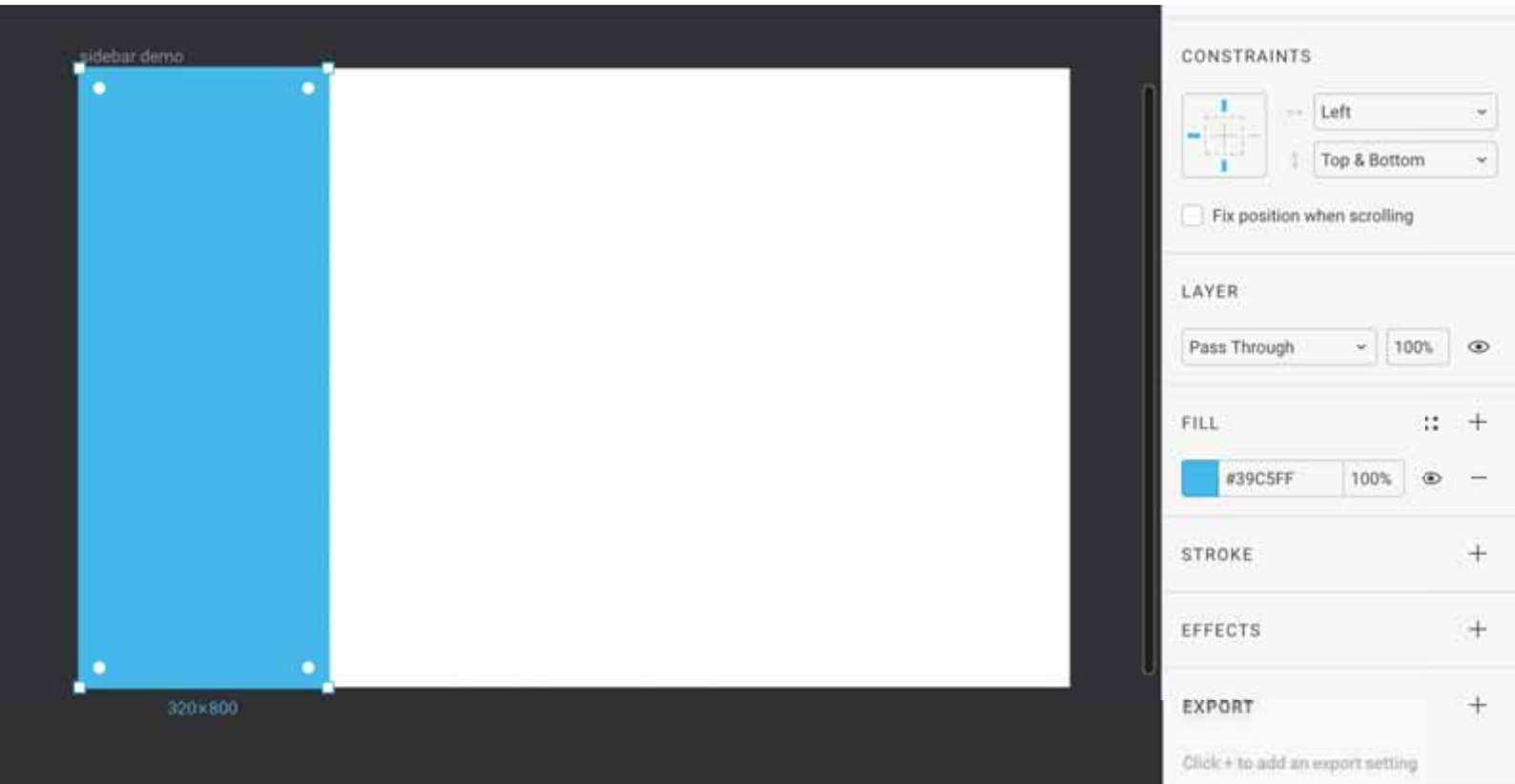
Как сделать сайдбар с резиновой высотой

Другой классический кейс использования ограничителей — интерфейс с боковой панелью — сайдбаром. В дизайне для веба высота страницы часто зависит от количества контента.

Задача: Привязать высоту сайдбара к высоте фрейма, чтобы можно было подстраивать высоту одним действием.

1. Создаём фрейм размером 1280 x 800, **F**.
2. Внутри него размечаем сайдбар размером 320 x 800, **R**. При создании он получает ограничители по умолчанию: **Left** и **Top**.

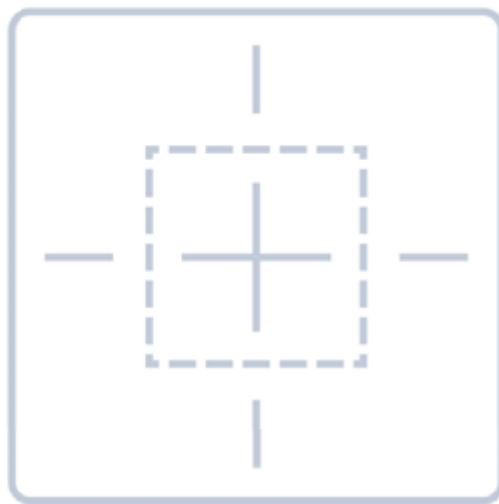
3. Зажимаем **Shift** и добавляем к ним ещё один ограничитель **Bottom**, кликнув на нижнюю вертикальную чётточку в блоке **Constraints**. Горизонтальный **Left** можно оставить, он пригодится, если будем делать макет адаптивным.



4. Тянем фрейм за нижний край, чтобы убедиться, что сайдбар тянется вслед за ним.

В Скетче настройка аналогичного поведения выглядит так:

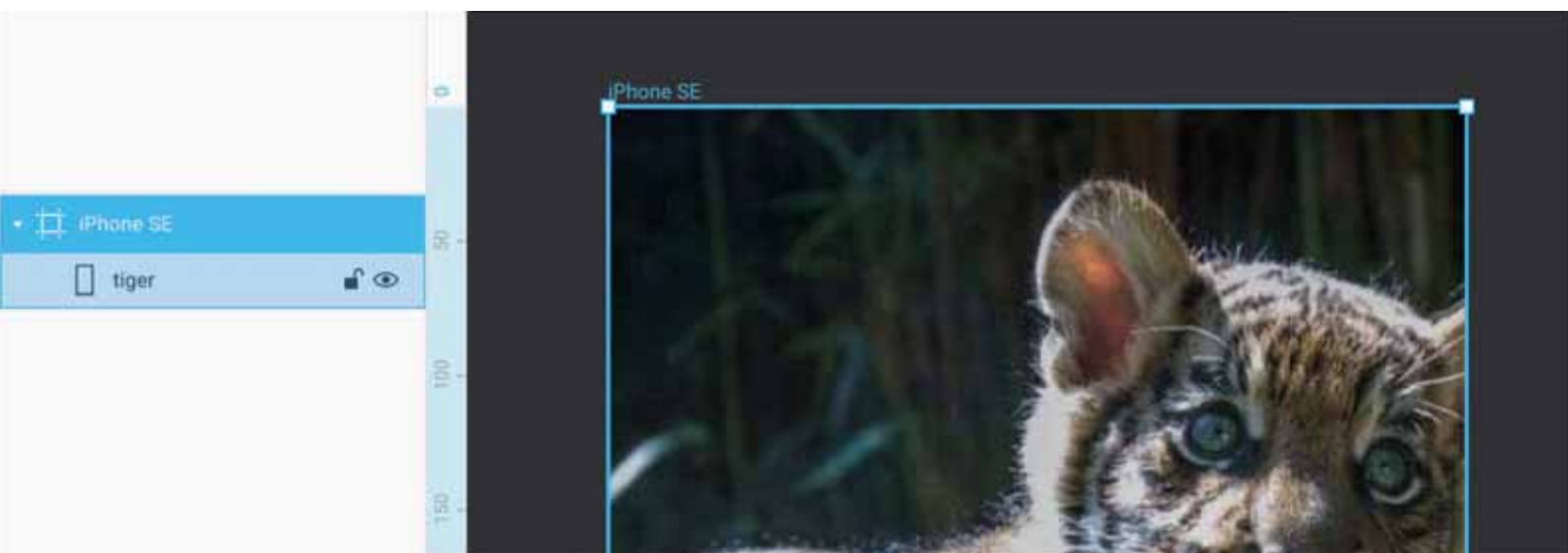




Как растянуть фоновое фото на весь фрейм

В качестве полноэкранного фона дизайнеры часто используют фотографии.

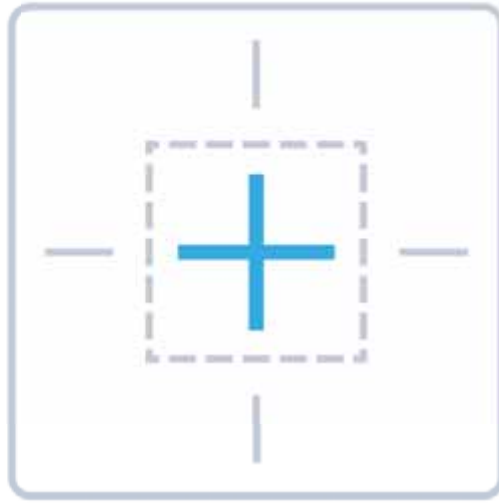
Сначала на ум приходит идея залить растровым фоном сам фрейм, но на практике это оказывается неудобно, потому что часто возникает потребность с клавиатуры копировать стиль заливки фона и переносить его в другие фреймы. Гораздо легче перетащить с рабочего стола фотографию в окно Фигмы и вписать её в фрейм, а затем снять на ней все ограничители.



Задача: при изменении размера фрейма фото должно подстраиваться.

1. Создаём фрейм, **F**. Выбираем размер iPhone SE.
2. Перетаскиваем с рабочего стола в Фигму файл фотографии в формате JPG или PNG.
3. Вкладываем слой изображения в фрейм, перетаскивая их друг на друга в панели слоёв:
4. Подстраиваем размер фотографии под размер фрейма, хватая за квадратные ручки в углах. Координаты фотографии — **0, 0**, размер: 320 x 568.
5. Жажимаем **Shift** и снимаем ограничители по умолчанию: **Left** и **Top**. Справа от прицела в меню горизонтальных и вертикальных ограничителей написано **Scale**. В этом режиме фото пропорционально тянется с двух сторон.
6. Меняем размер артборда на iPhone 8 и убеждаемся, что фото растянулось.





Как зафиксировать модальное окно по центру экрана

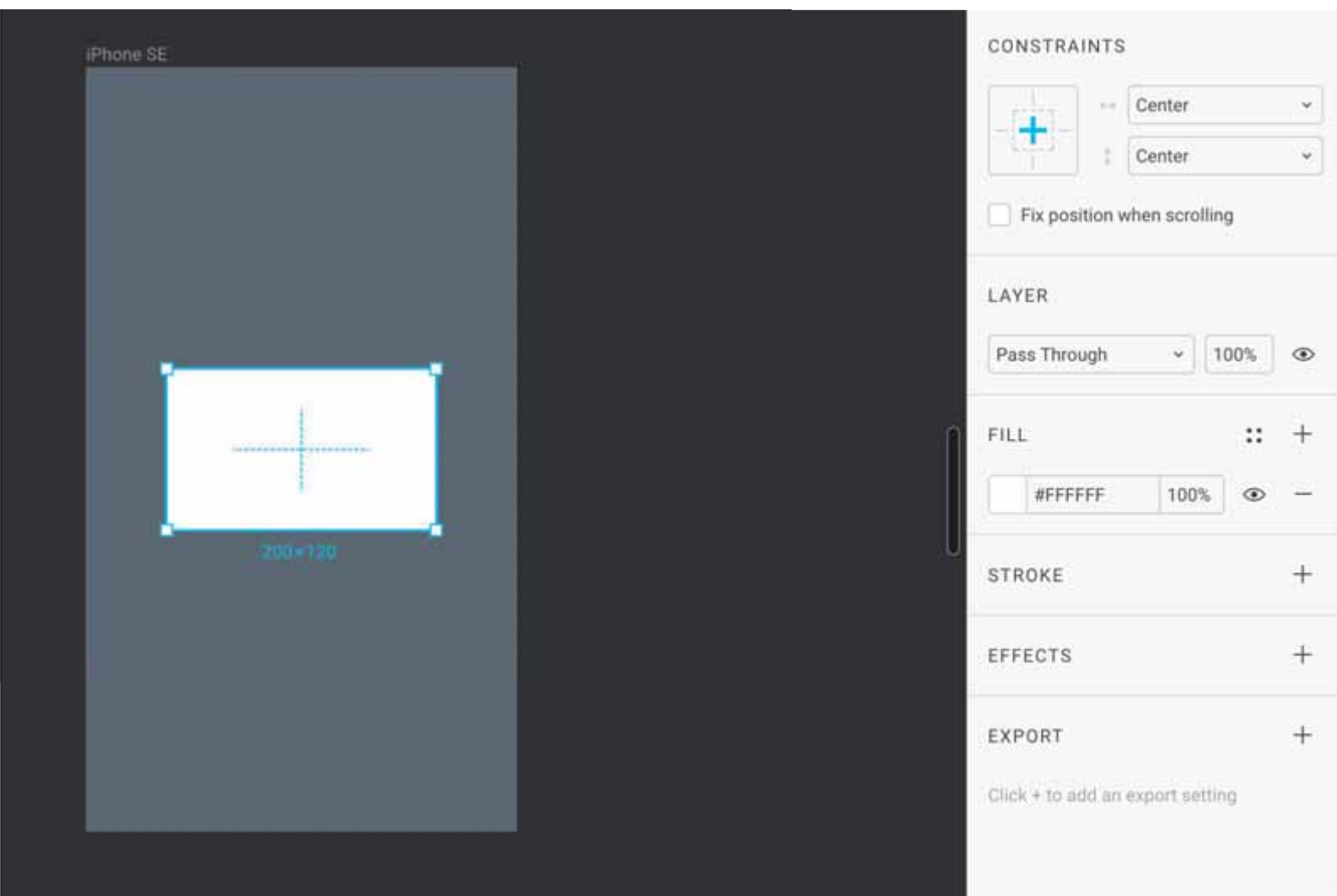
Функции выравнивания **Horizontal Center**, **Opt + H** и **Vertical Center**, **Opt + V** позволяют двигать слой на центр его фрейма. Если тот изменится, слой останется на тех же координатах.

Если мы заранее знаем, что слой всегда должен быть строго по вертикальному или горизонтальному центру, вне зависимости от размера фрейма, можно использовать ограничители и прикрепить слой к центру.

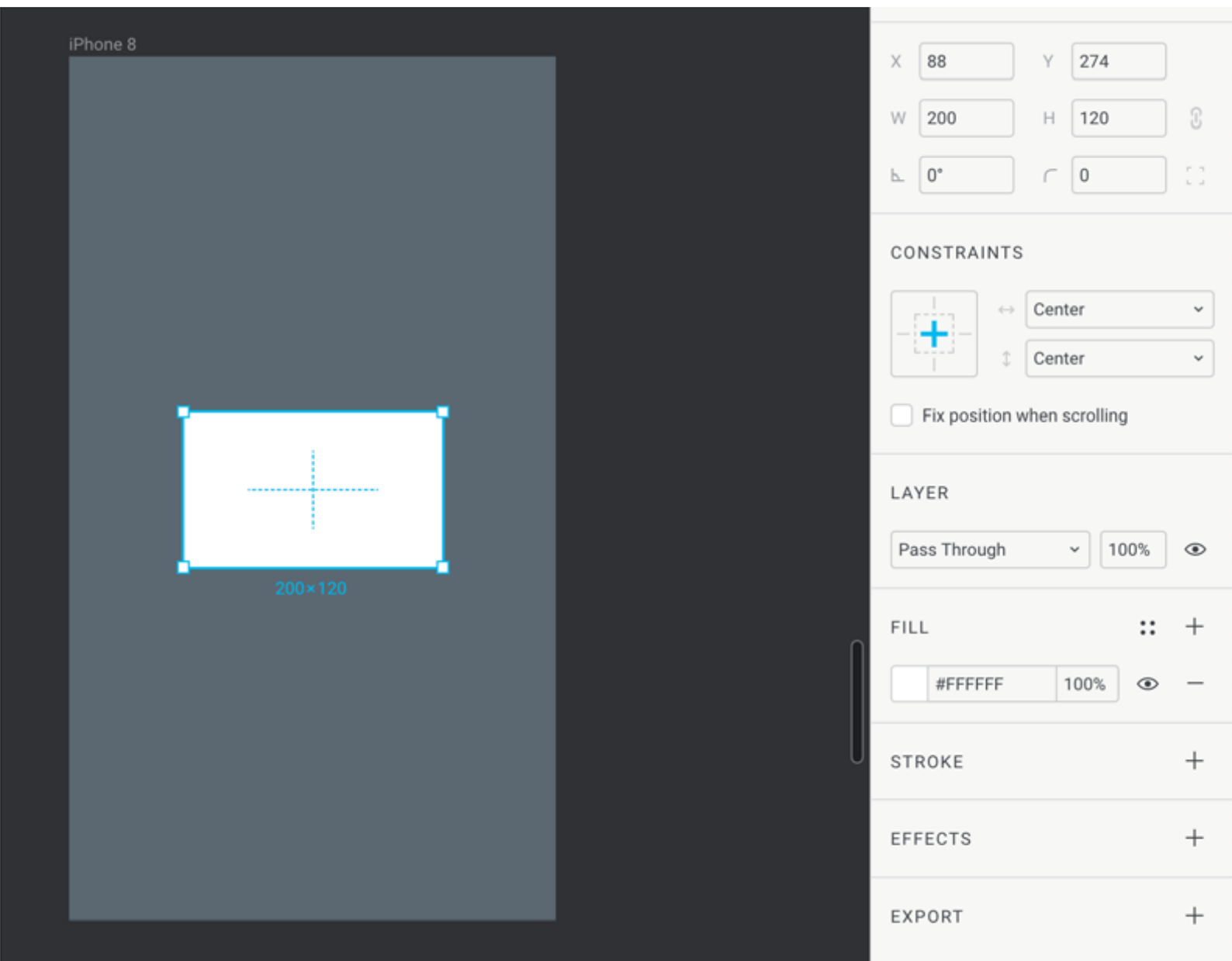
Задача: модальное окно должно быть по центру

1. Создаём фрейм размера **iPhone SE, F**.
2. Рисуем прямоугольник размером 200 x 120, **R**.
3. Ставим его по центру фрейма, **Opt + H, V**. Нажимаем **V** после **H**, не отпуская **Opt**.
4. Когда зажат Opt, мы можем смотреть расстояния от прямоугольника до краёв фрейма.
5. Настраиваем прицел в блоке **Constraints** так чтобы в нём был голубой плюс, кликнув на каждую из перекладин. Либо выберем из меню значения **Center, Center**.

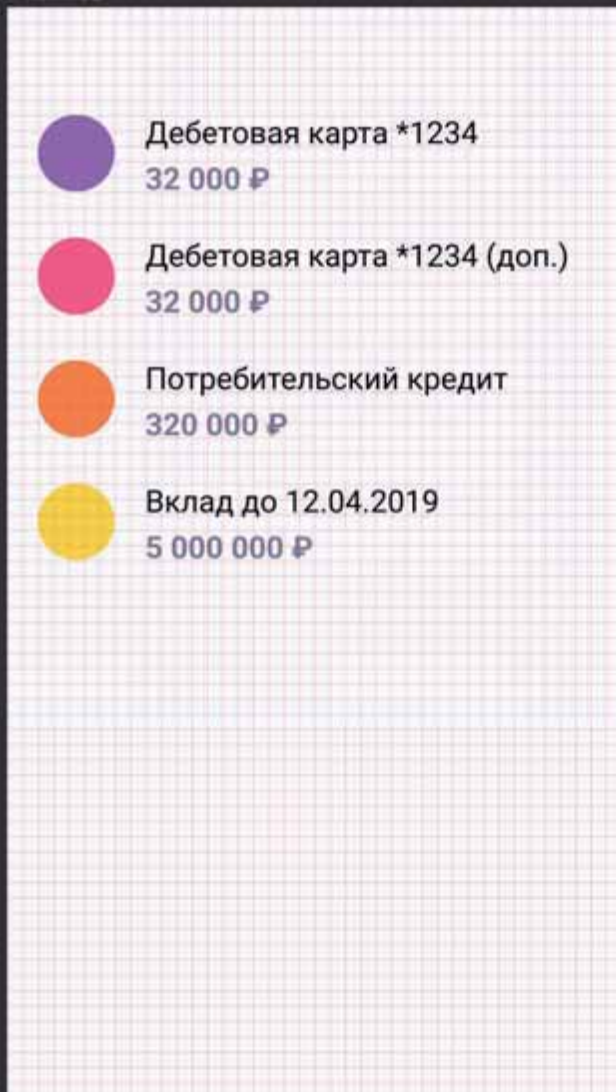
Обратим внимание, что внутри прямоугольника появился пунктирный крест, ещё одно подтверждение того, что мы в режиме **Center, Center**.



6. Сменим размер фрейма с **iPhone SE** на **iPhone 8**. Прямоугольник подстроит своё положение:



Закреплённый слой можно сдвигать относительно центра. При изменении фрейма тот сохранит своё расстояние сдвига в пикселях. Если слой сдвинули вверх на 10 пикселей и изменили фрейм, в новом фрейме это расстояние не исказится.



Все мы были в этой неловкой ситуации...

Как сделать так, чтобы при растягивании макета всё не ехало

Перед началом растягивания зажать **Cmd** на Маке или **Ctrl** на Windows.

Отключить все ограничители галочкой как в Скетче нельзя. Однако, при помощи этой клавиши можно временно их игнорировать. Это удобно, если в макете много слоёв, ограничители не настроены и прописывать правила поведения слишком долго.



20. Текстовые слои

[Проект в Фигме →](#)

80% дизайна интерфейсов — это работа с типографикой.

В Фигме есть хорошие возможности для работы с текстом через специальный тип слоёв.

Чтобы создать текстовый слой, нажимаем **T** и кликаем в нужное место, либо растягиваем блок как прямоугольник или фрейм.



BYOF (Bring your own fonts!)

Figma comes preloaded with Google Web Fonts. To use your local fonts as well, you'll need our font installer. Once installed, you'll always have access to your local fonts when working on this computer.

You can always download the font installer later from the settings page.

Do this later in account settings

Enable local fonts



Саша Окунев

Шрифты из Google Web Fonts

Хорошая новость: в Фигме изначально доступна огромная библиотека шрифтов [Google Fonts](#), которую удобно использовать в веб-разработке. Многие из доступных шрифтов поддерживают кириллицу. Если ограничен бюджет проекта, рекомендую использовать в дизайне те шрифты, которые доступны в этой платформе.

Повторю первую главу: чтобы локальные шрифты были доступны в браузере, нужно установить десктопный клиент, либо запустить [Font Helper](#).

В Скетче текстовый слой может быть в двух режимах:

Auto и **Fixed**. В Фигме таких режимов три:

- **Width** – полный аналог режима **Auto**
- **Height** – нет в Скетче
- **Fixed** – в Скетче называется и работает так же.



Булочка

63×19

TEXT

Roboto

Bold

16

A

100%

|A|

0%

A

0

→

0



Создадим текстовый слой. Если выделен текстовый слой, справа мы увидим панель работы с текстом.

Справа вверху блока иконка **Text Styles** — всплывающее меню стилей.

Далее идут поля:

- **Typeface**
- **Weight** — жирный **Cmd + B** и курсив **Cmd + I**
- **Size**
- **Line Height** — высота строк
- **Letter Spacing** — трекинг
- **Paragraph Spacing** — отступ между параграфами, разделёнными обрывом строки, **Enter**
- **Paragraph Indentation** — красная строка

Далее идёт настройка выравнивания по вертикали и горизонтали:



Opt + Cmd + L	Text Align L eft
Opt + Cmd + T	Text Align Center t er
Opt + Cmd + R	Text Align R ight
Opt + Cmd + J	Text Align J ustify

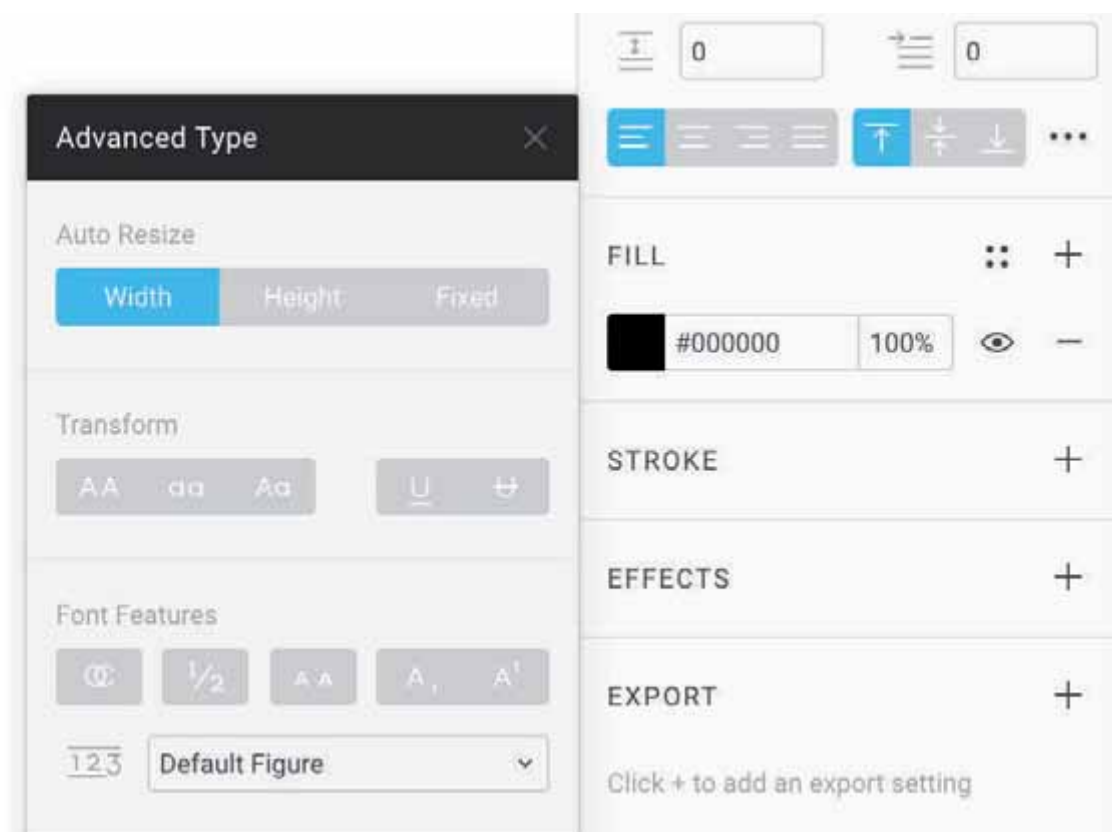
За неимением нормальных клавиш для вертикального выравнивания, можем находить команды через поиск:

Cmd + /, te to	T ext Align T op
Cmd + /, te mi	T ext Align M iddle
Cmd + /, te bo	T ext Align B ottom

Напротив выравнивания — меню **Advanced Type** (три точки).

Auto Resize: Width

Если текстовый слой был создан кликом, он имеет настройку **Auto Resize** в значении **Width**, и его ширина подстраивается под контент. Длинный текст развяжется в одну строку.



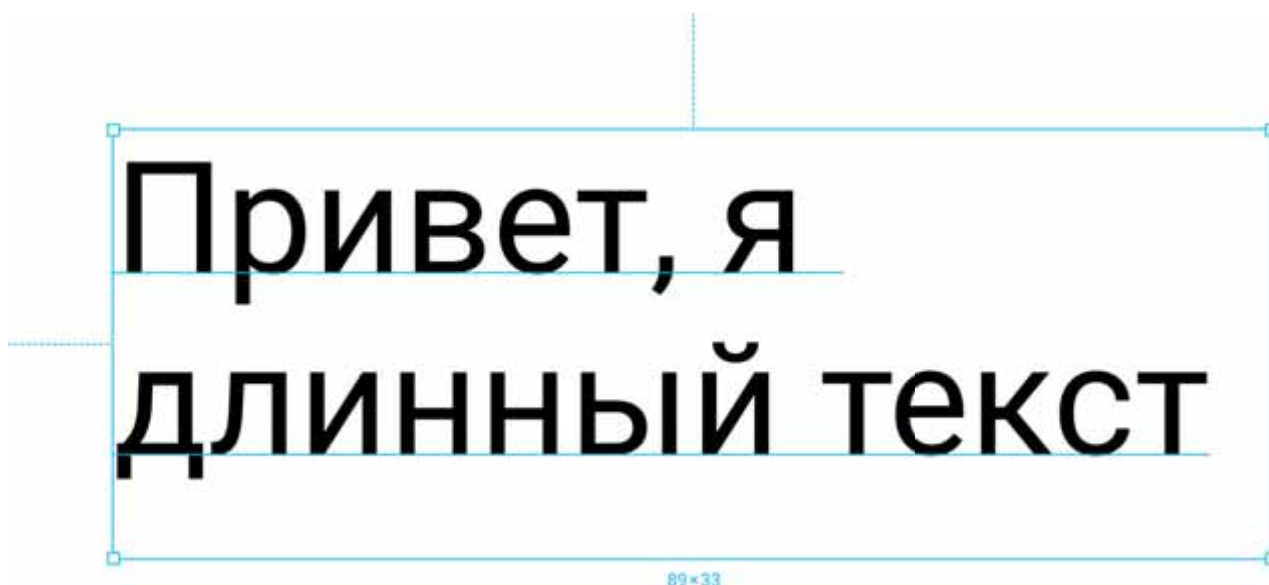
Если такую строку потянуть за правый край, настройка **Auto Resize** переключится в режим **Fixed**. Это означает, что текстовый блок будет иметь фиксированную ширину.

Если слой изначально создан перетаскиванием, текстовый блок сразу будет иметь фиксированную ширину. Такие слои удобно использовать в компонентах, где мы заранее не знаем ширину контента и задаём максимально допустимую ширину.

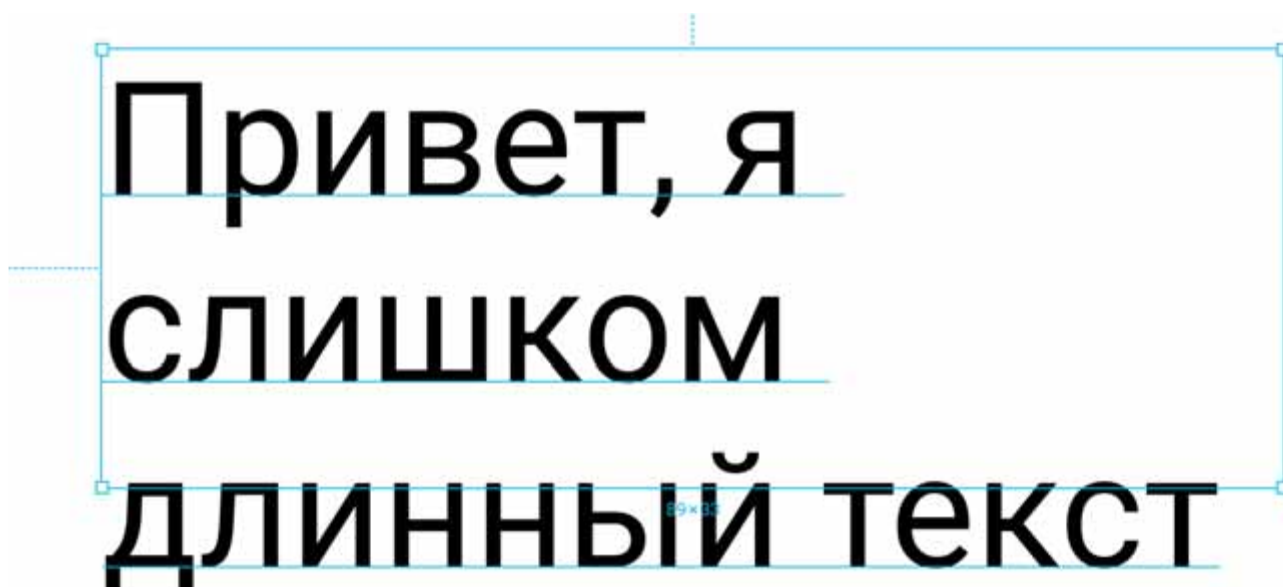
Auto Resize: Height

Третий тип авторесайза, которого нет в Скетче. Позволяет подстраивать высоту текстового блока под его контент.

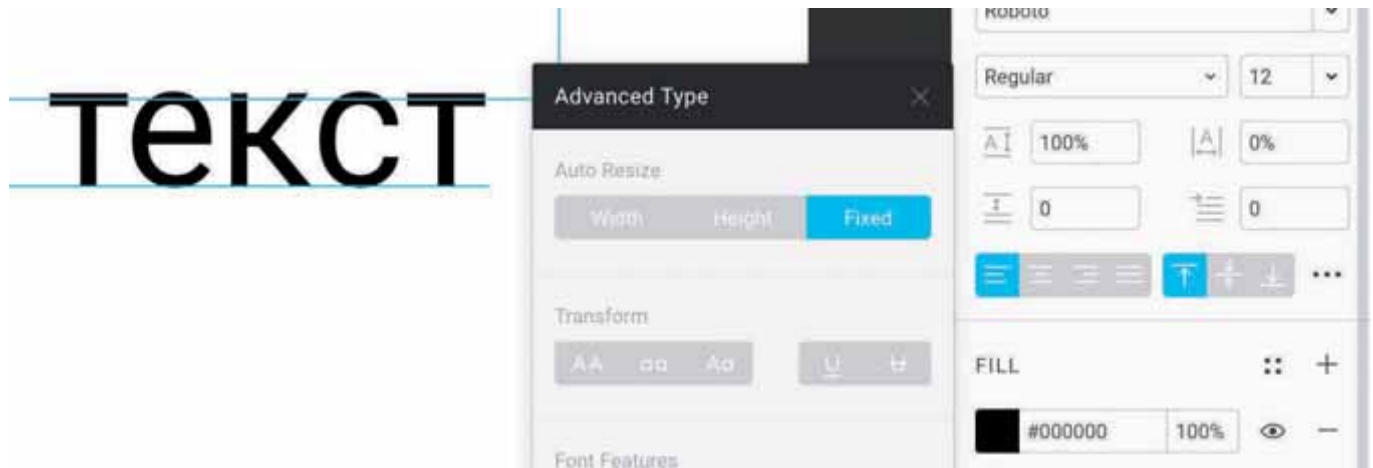
Допустим, мы создали текст с фиксированной шириной и высотой. Выбрали **Text Tool**, **T**, растянули бокс и напечатали:



Всё хорошо, пока строки укладываются в бокс. Но если текст становится слишком длинным, он перестаёт укладываться в высоту блока и некрасиво свисает:



Чтобы этого не происходило, нужно переключать режим текстового блока в **Height**. Чтобы сделать это, заходим в меню **Advanced Type**, нажав на три точки в блоке **Text**.

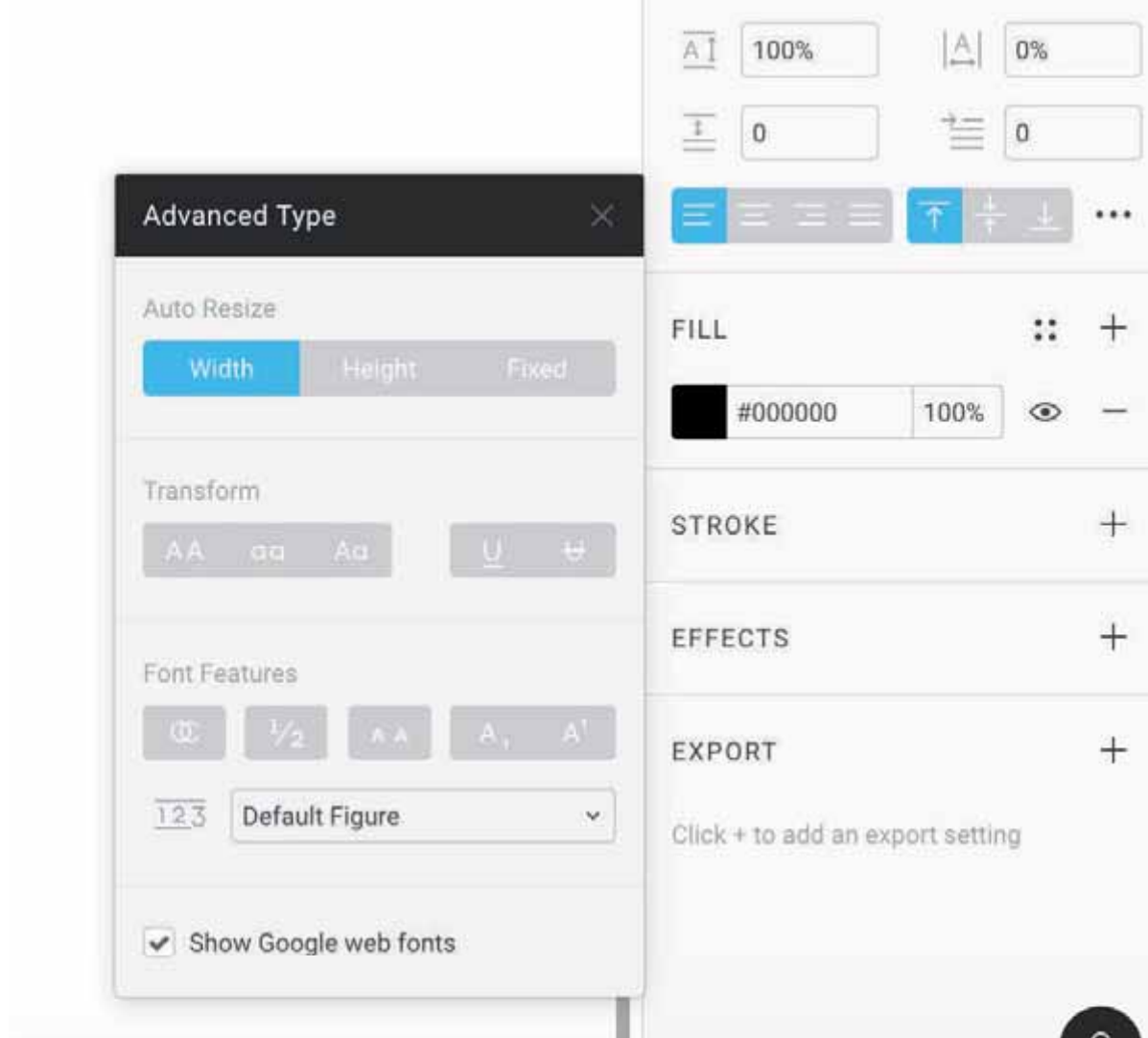


После этого бокс будет подстраиваться под размер текста по высоте.

Также в панели **Advanced Type** можно настроить дополнительные свойства текста:

Transform — трансформация текста в заглавные или строчные

Underline и **Strikethrough** [страйктуру] — подчёркнутый и зачёркнутый текст.



Блок Font Features

Содержит дополнительные возможности шрифта. Иными словами, никогда не используемые дебри.

Discretionary and historical ligatures — использовать ли дискретные и исторические [лигатуры](#). О возможностях OpenType шрифтов есть [жирная статья на ilovetypography.com](#) (англ). Мало какие шрифты поддерживают эту функцию.

Fractions — дроби. Соединять ли такие значения как 1/2 в один глиф с дробью.

21. Выравнивание и распределение

[Проект в Фигме →](#)

Эта глава о базовых способах выравнивания слоёв в макетах и о умном распределении карточек в пространстве.

Align: Выравнивание

Чтобы выравнивать по фрейму, выделяем квадрат кликом.

 **Opt + A** **Align Left.** По левому краю.
Также работает **Ctrl + Cmd + ←**.

 **Opt + H** **Align Horizontal Centers.**
По горизонтальному центру.

 **Opt + D** **Align Right.** По правому.
Также работает **Ctrl + Cmd + →**.

 **Opt + W** **Align Top.** По верхнему.
Также работает **Ctrl + Cmd + ↑**.

 **Opt + V** **Align Vertical Centers.**
По вертикальному центру.

 **Opt + S** **Align Bottom.** По нижнему.
Также работает **Ctrl + Cmd + ↓**.

Полезно запомнить комбинированную клавишу **Opt + H, +V**, которая позволяет выравнивать слой по центру фрейма.

Distribute: Распределение

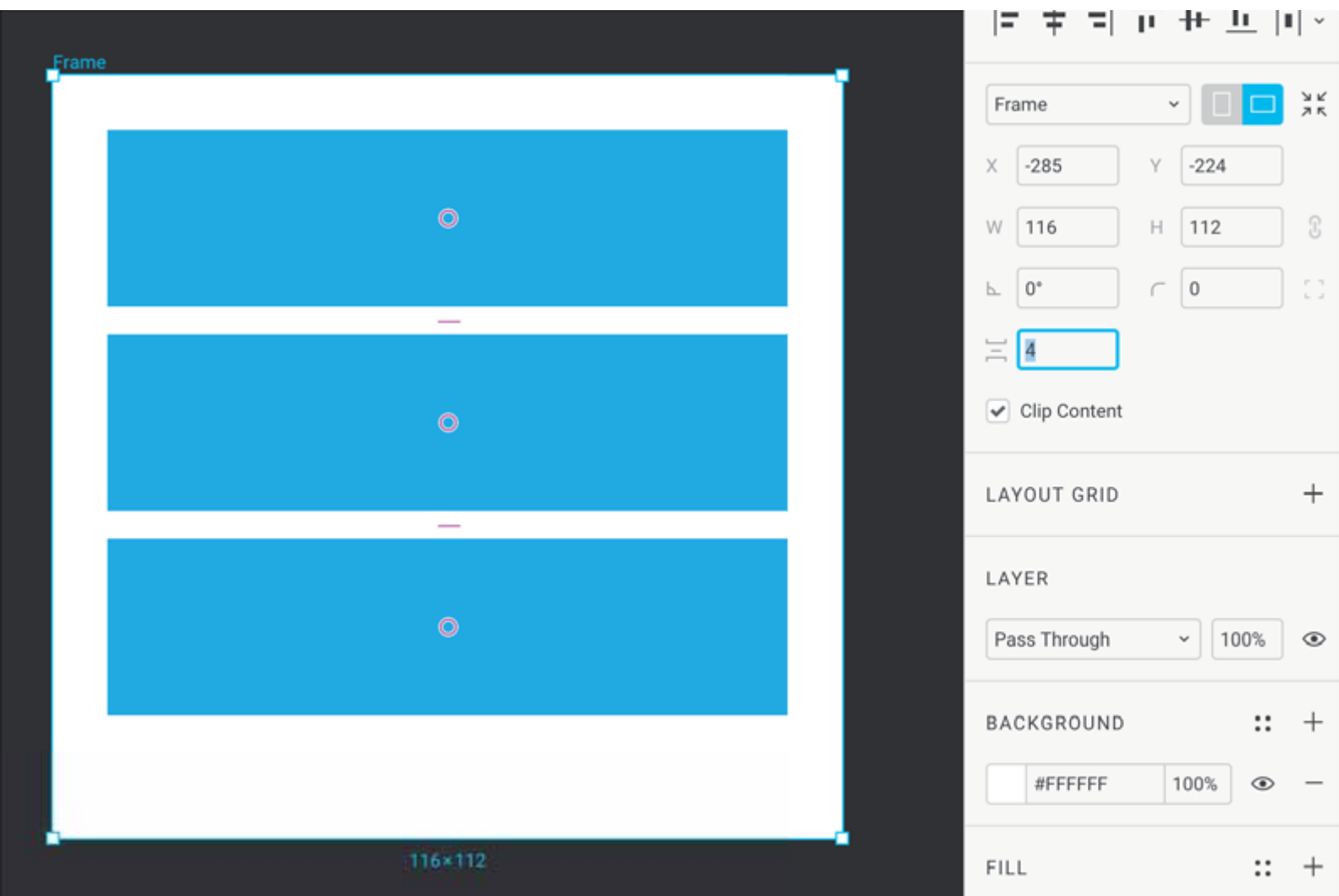
Фигма вводит новый стандарт работы со списками карточек и таблиц. В Скетче нет ничего подобного. Там выравнивать карточки можно плагинами, но на них полагаться рискованно, поскольку каждую новую версию Скетча они ломаются.

Чтобы распределять, выделяем несколько фигур, минимум 3.



Ctrl + Opt + T

Tidy Up. Умное распределение позволяет задать одно и то же расстояние между фигурами. Например, если в дизайне используются карточки в столбик, им можно задать одинаковый отступ, который удобно менять в поле отступа.



Ctrl + Opt + V **Distribute Vertical Spacing.**

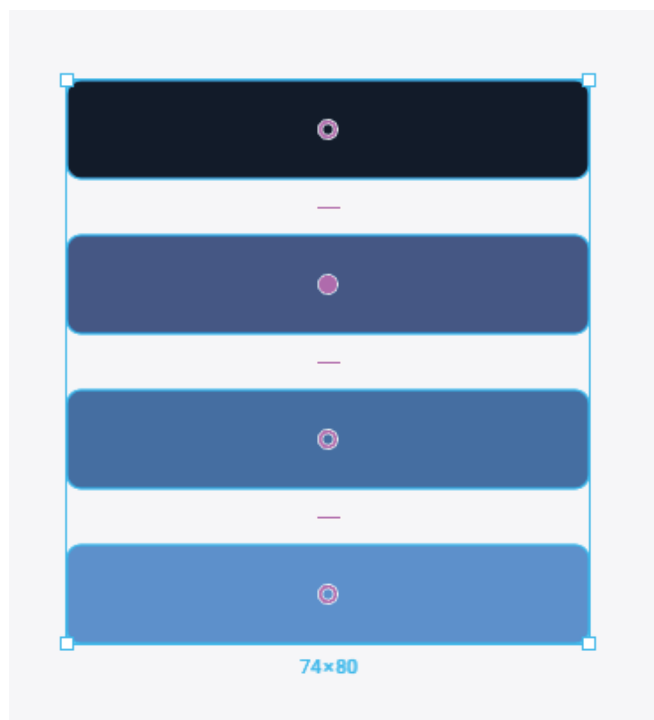
Выравнивает по верхнему объекту, задаёт равные вертикальные отступы в стопке слоёв. Если применить его, во вкладке **Design** появляется дополнительное поле, куда можно вписать желаемый отступ.

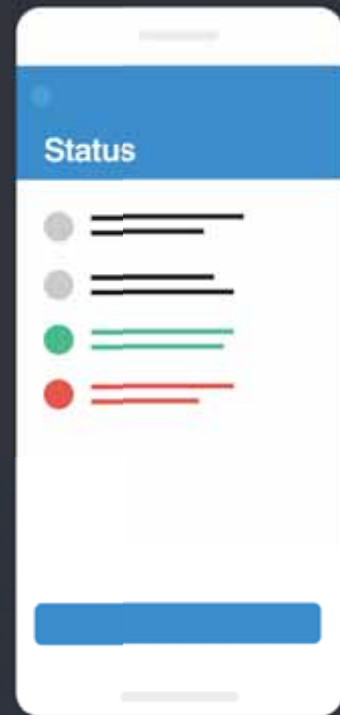
Shift + ↑ / ↓ если выделено поле отступа, позволяет менять размер отступа по всей выделенной группе объектов с увеличенным шагом.

Фиолетовые рули в центре позволяют сортировать последовательность блоков перетаскиванием вверх-вниз.

Ctrl + Opt + H **Distribute Horizontal Spacing**

Работает аналогично **Vertical Spacing**.





22. Стили цветов

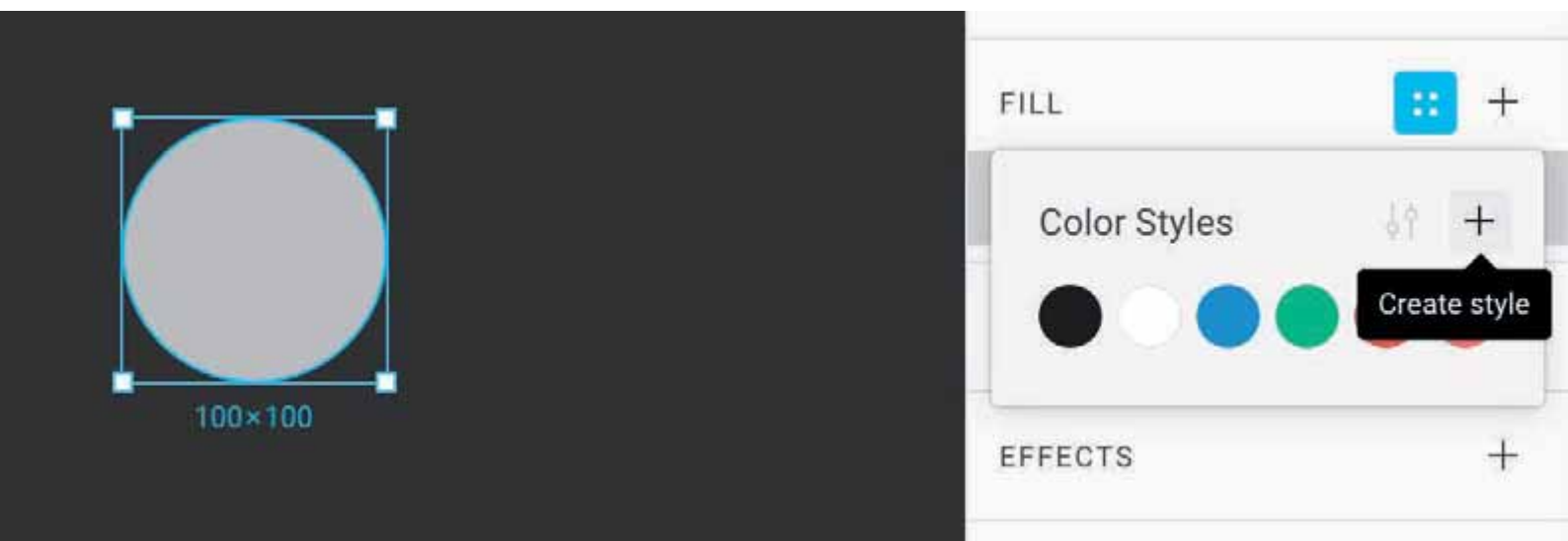
[Проект в Фигме →](#)

Мы используем функцию **Color Styles**, когда нужно добиться единства в цветах по всему проекту. Они позволяют задавать основную палитру проекта и назначать именованные цвета, которые можно использовать для заливки, в обводках и тексте.

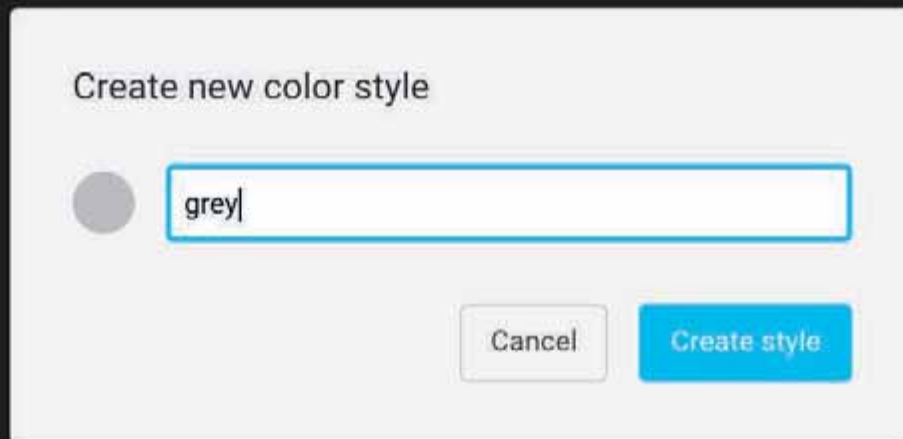
Впоследствии цвет в стиле можно заменять по всему проекту, редактируя стиль.

Задача: создадим простейшую палитру проекта и подвяжем стили к слоям.

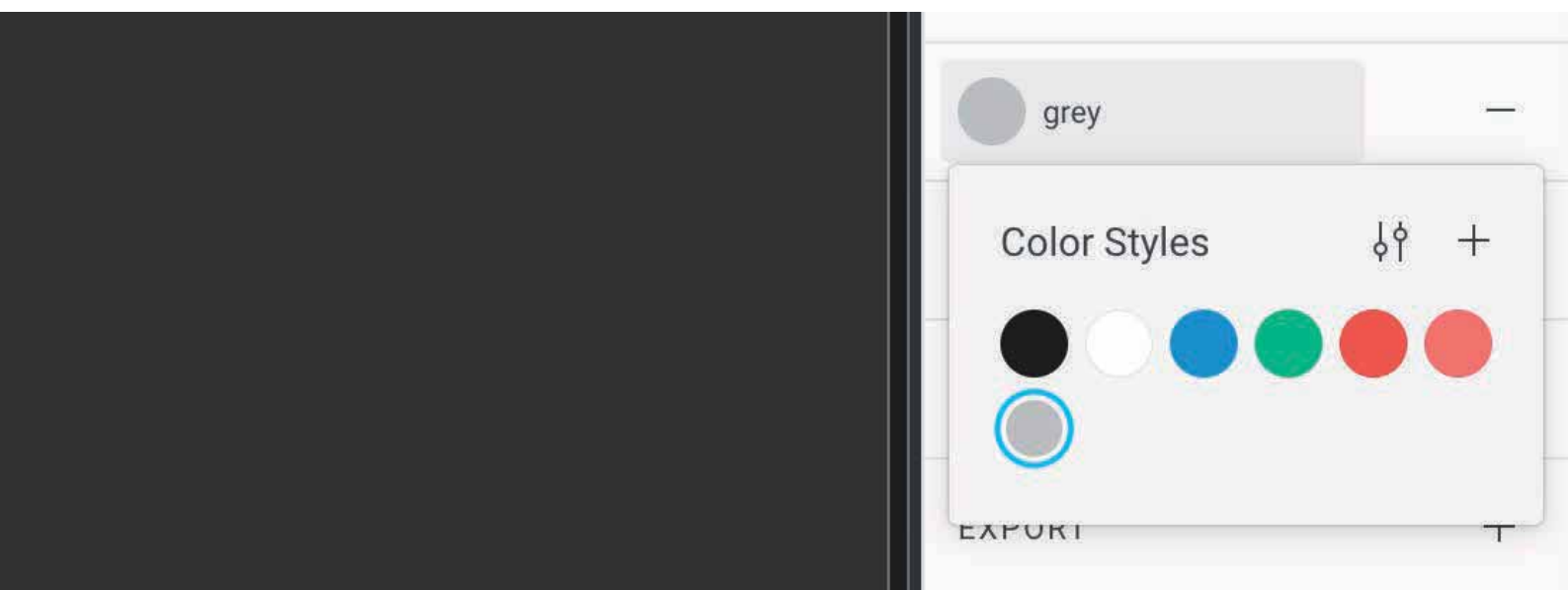
1. Создаём круг, **O**. Он будет пробником, в котором будем подбирать цвет.
2. Выбираем нужный цвет или оставляем серый. Он будет тем цветом, который мы пропишем в стиль.
3. В блоке **Fill** находим иконку с четырьмя точками. Клик по ней раскрывает панель **Color Styles**.
4. Нажимаем кнопку +, **Create Style**.



5. Заполняем имя цвета, желательно латиницей и без пробелов. Вместо них можно использовать - или camelCase. Это нужно, потому что впоследствии твои разработчики будут так называть переменные и было бы круто, если бы стили цветов и названия переменных синхронизировались.



6. Стиль создан. Теперь круг залит серым цветом стиля **grey**.



Копируем стили с одного объекта на другой

7. Создаём ещё один круг, **O**. Меняем ему цвет, чтобы был не серый. Изначально у него нет стиля, а есть однотонная заливка.



8. Выделяем первый круг, копируем его стиль командой **Copy Properties**, **Opt + Cmd + C**.
9. Выделяем второй круг, вставляем скопированный стиль второго командой **Paste Properties**, **Opt + Cmd + V**. Он становится серым, а блок **Fill** в панели свойств пропадает. Теперь ко второму слою подвязан стиль **grey**.



Сходства и различия стилей в Фигме и Скетче

Реализация стилей в Фигме отличается от Скетча в лучшую сторону.

Сравним их.

Скетч: стиль хранит все настройки шейпа

В Скетче палитру можно контролировать через стили шейпов – **Layer Styles**. В стиль попадают все его настройки:

- Цвет и толщина обводки
- Цвет заливки
- Настройки теней
- Опасность

Если обводок или заливок несколько, они тоже попадут в стиль.

Когда это неудобно: Например, возникает потребность сделать определённый шейп полупрозрачным. Связь между исходным стилем и видоизменённым будет разорвана. Настройки слоя больше не соответствуют исходному стилю, о чём сигнализирует звёздочка после его названия в блоке **Appearance** [эпиранс]:

При изменении исходного стиля настроенная полупрозрачность слетит и это будет трудно отследить.

Чтобы не наступить на эти грабли, приходится делать два независимых стиля, даже если оба шейпа должны быть одного цвета и разница только в значении опасности.

С одной стороны, подход Скетча позволяет тонко контролировать все свойства шейпов в проекте. С другой, это никому не нужно.

Цена порядка слишком высока. Такая строгость приводит к тому, что стили множатся с удручающей скоростью, капризны в настройке, их сложно удерживать в порядке, потому что окно работы со стилями **Organize Layer Styles** не даёт никакой информации о их настройках. До версии Скетча 53 я вообще не использовал стили шейпов, потому что поддерживать вид слоёв в актуальном состоянии было дешевле через символы с прямоугольниками-цветами.

Фигма: стиль — это универсальная настройка цвета, применимая к любым типам слоёв: шейпам, текстам, векторам и фреймам.

В стиле хранится только цвет. Это значит, его можно применить к заливке, обводке и на текстовых слоях. Такой подход позволяет контролировать цвет в проекте вне зависимости от того, где он использовался: в заливке блока, линиях или в текстах.

Если в Фигме нужно сделать шейп полупрозрачным, это не нарушит его связь со стилем.

Цвета можно называть и комментировать

Даже если два стиля содержат очень похожие цвета, можно избежать путаницы, давая им уникальные названия. Принято использовать латиницу без пробелов: **green, blue, grey-dark**. Такие названия разработчикам будет проще вносить в переменные.

При наведении на образец цвета его название всплывает в контекстной подсказке.



Header



23. Компоненты

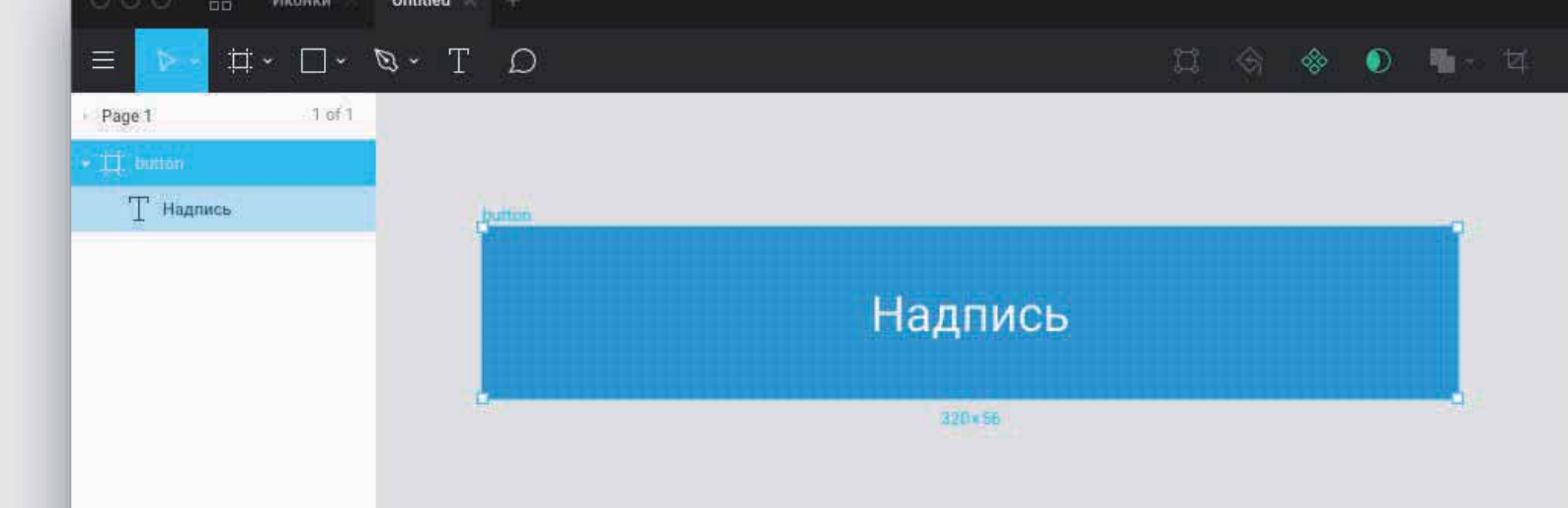
[Проект в Фигме →](#)

Важнейшая тема, которая лежит в основе дизайн-систематизации и позволяет делать сложные интерфейсы, которые реально поддерживать в актуальном состоянии. Компоненты в Фигме — аналог символов в Скетче и классов в программировании.

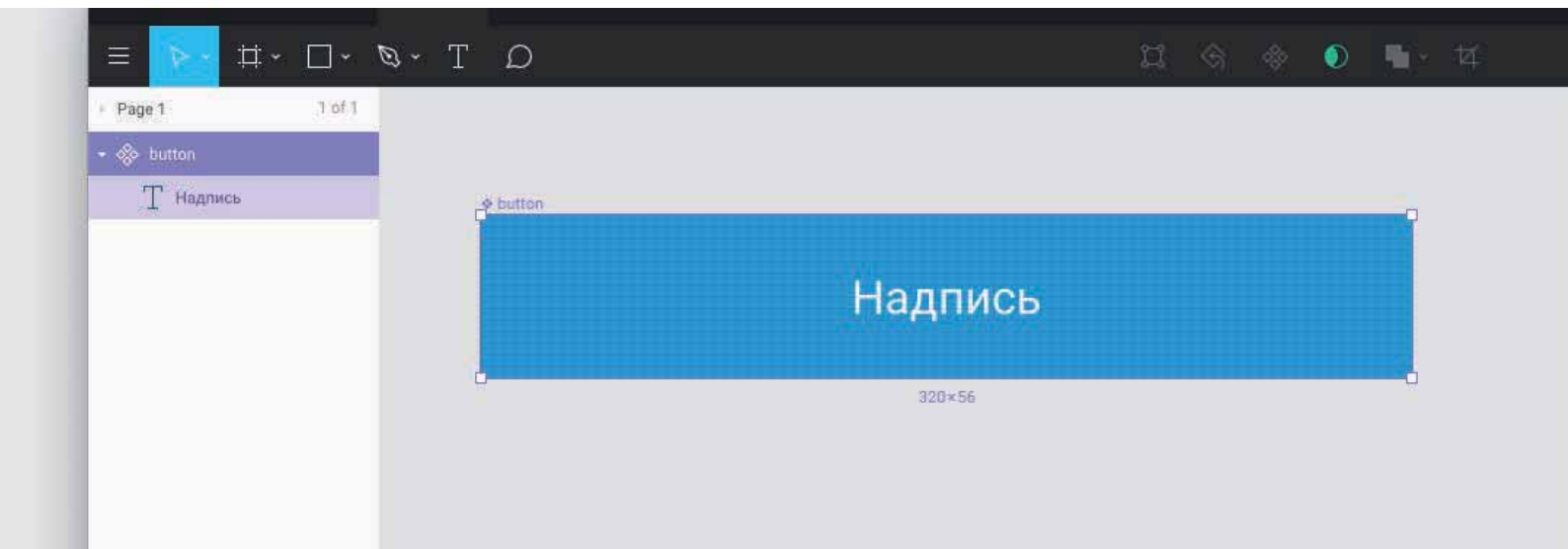
Компоненты можно создать в одном месте, а затем использовать в других. В основе любого компонента есть фрейм. Его содержимое служит основой дальнейших копий, называется **мастером** или родителем. Все копии с этого слоя — **экземплярами** или детьми.

Сделаем компонент кнопки

1. Сделаем прямоугольник и текстовый слой внутри. Чтобы создать компонент, выделяем объект и нажимаем **Opt + Cmd + K**.



Мнемоника: **К**омпонент. Фрейм становится фиолетовым и превращается в мастер-компонент:



Запомним: фиолетовый в Фигме — цвет компонентов.

Немного о клавишах

Создание компонентов — одна из команд, которую обычно дают с клавиатуры, поэтому **Opt + Cmd + K** важно запомнить.

В Скетче для создания символа я назначал клавишу **Cmd + Y**.

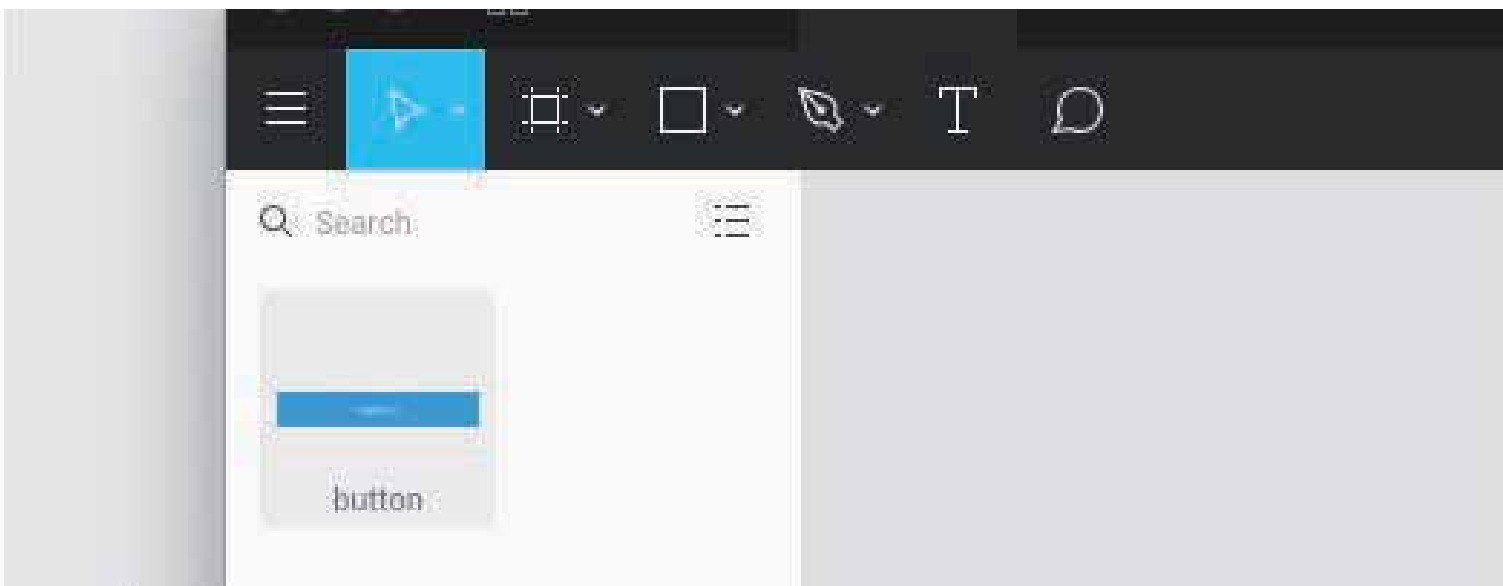
Мнемоника: **Symbol**. Но в Фигме на эту клавишу уже назначено другое полезное действие и усложнять не хочется. Какие сочетания использовать — личное дело каждого. Бывает, что дизайнер вынужден работать в обоих редакторах, постоянно переключаясь между ними. В этом случае я бы назначил на одно действие одну и ту же клавишу. Тогда получится работать с редактором на моторном уровне, минимально задумываясь, какой из них открыт в данный момент.

Сравнение символов Скетча

и компонентов Фигмы

В Скетче мастер-символы являются артбордами. По аналогии, в Фигме мастер-компоненты являются фреймами.

Теперь созданный ранее компонент **button** доступен во вкладке **Components**, **Opt + 2**. Из неё экземпляры компонентов можно переносить на холст.

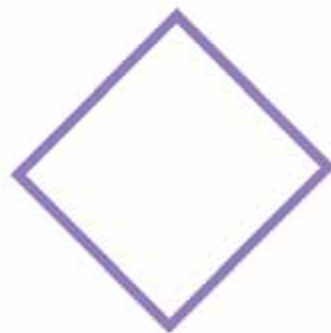


Также можно выделять экземпляры из мастера, зажав **Opt** и перетаскивая левой.





Мастер



Экземпляр

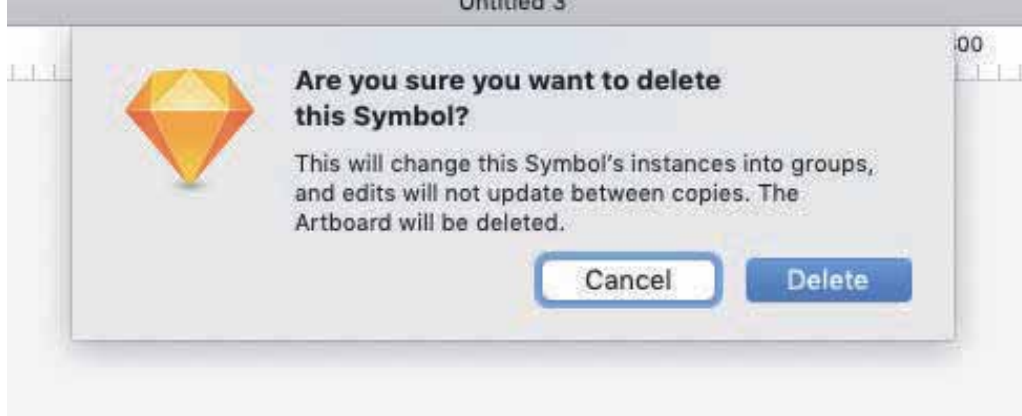
Мастеры: сходство

Мастер-компонент и мастер-артборд — прототип объекта, из которого происходят экземпляры. Им можно перезаписывать свойства, стили и контент.

Мастеры: различия

Скетч: когда объект становится символом, мастер-артборд создаётся на странице **Symbols**. Если в процессе создания символа сняли галочку **Send to Symbols page**, символ появляется в неконтролируемом месте той же страницы.

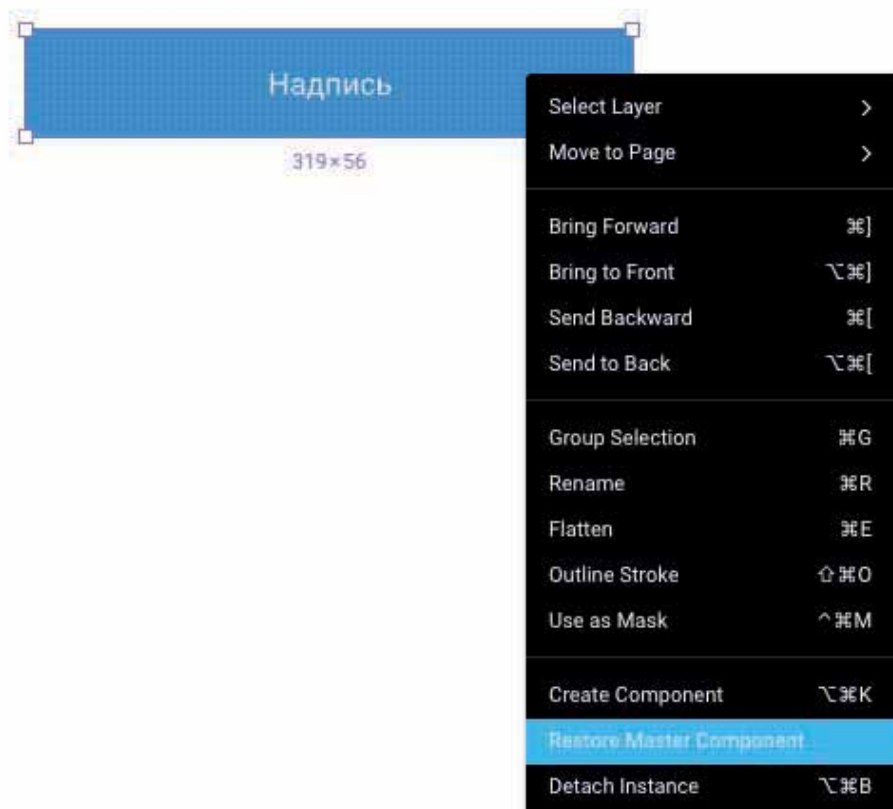
Фигма: когда объект становится мастер-компонентом, он остаётся на том же месте где был. Внешне он отличается от экземпляра только иконкой. Ничего не мешает использовать мастер-компонент в качестве экземпляра, но этого не стоит делать.



Различие в логике дублирования

Скетч: Если дублировать артборд мастера, **Cmd + D**, мы получим ещё один мастер с другим внутренним идентификатором. Вставить экземпляр можно только через меню **Insert** или через плагин **Runner**.

Фигма: Если дублировать фрейм мастера, **Cmd + D**, мы получим экземпляр этого мастера. Он встанет поверх мастера.



Различие в логике детача

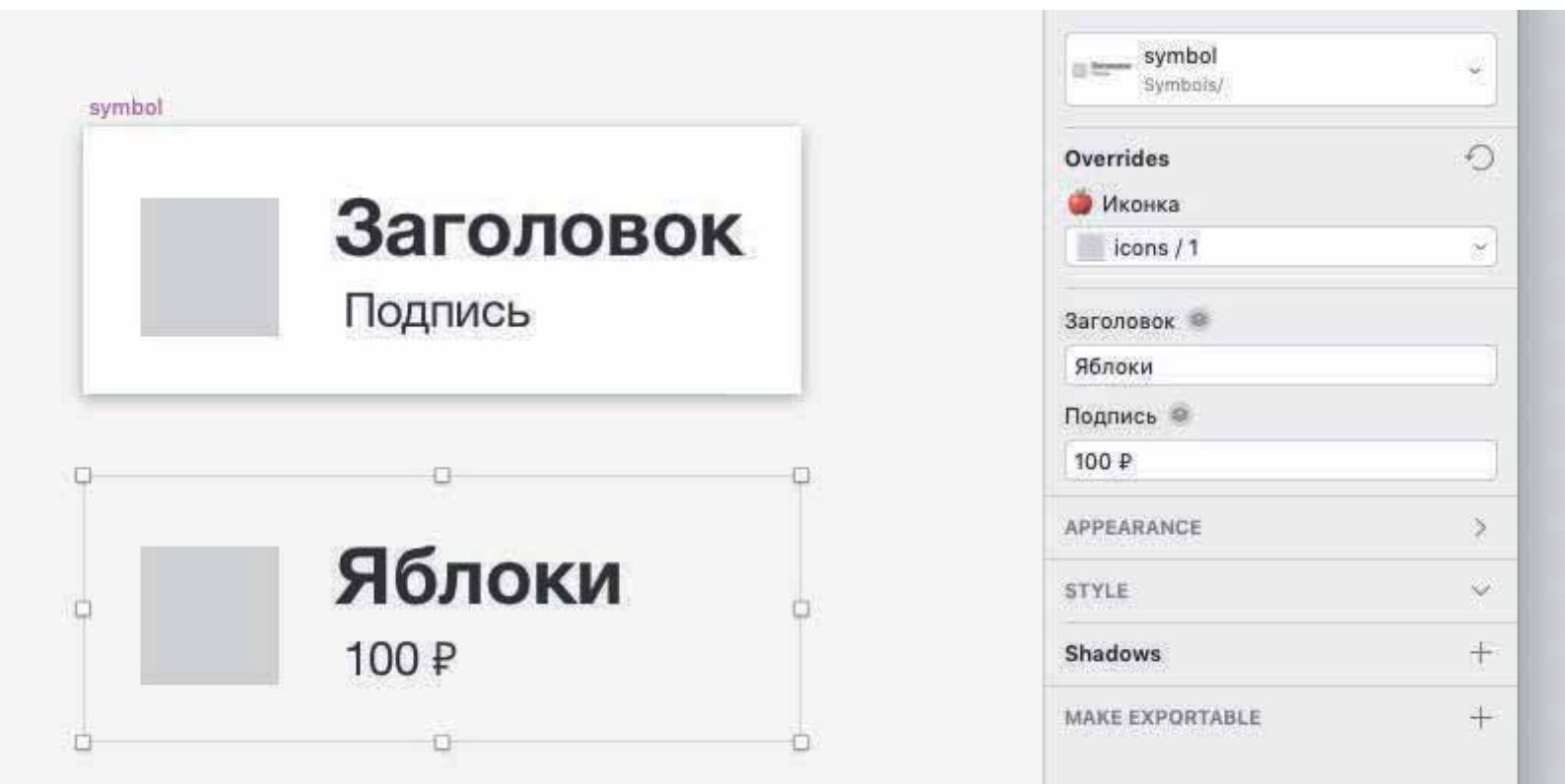
Скетч: когда мы пытаемся удалить мастер-артборд, происходит проверка, есть ли у него живые экземпляры, использующие его в качестве родителя. Если есть, мы видим меню **Are you sure you want to delete this Symbol**. Если нажимаем **Delete**, все экземпляры потеряют связь с мастером и остаются группами.

Фигма: Если удалить мастер, никакого окна о подтверждении не будет. Экземпляры после удаления сохраняют информацию о мастере. Если вырезать или удалить мастер, экземпляр останется экземпляром без явного мастера. В любой момент его можно восстановить в контекстном меню: **Restore Master Component**, **Cmd + /, resto**.

Мастер восстановится на тех же координатах, где он был.

Различие в реализации оверрайдов

Скетч: оверрайды можно перезаписывать только в панели **Inspector** справа. Контент отделён от слоёв в рабочей области.



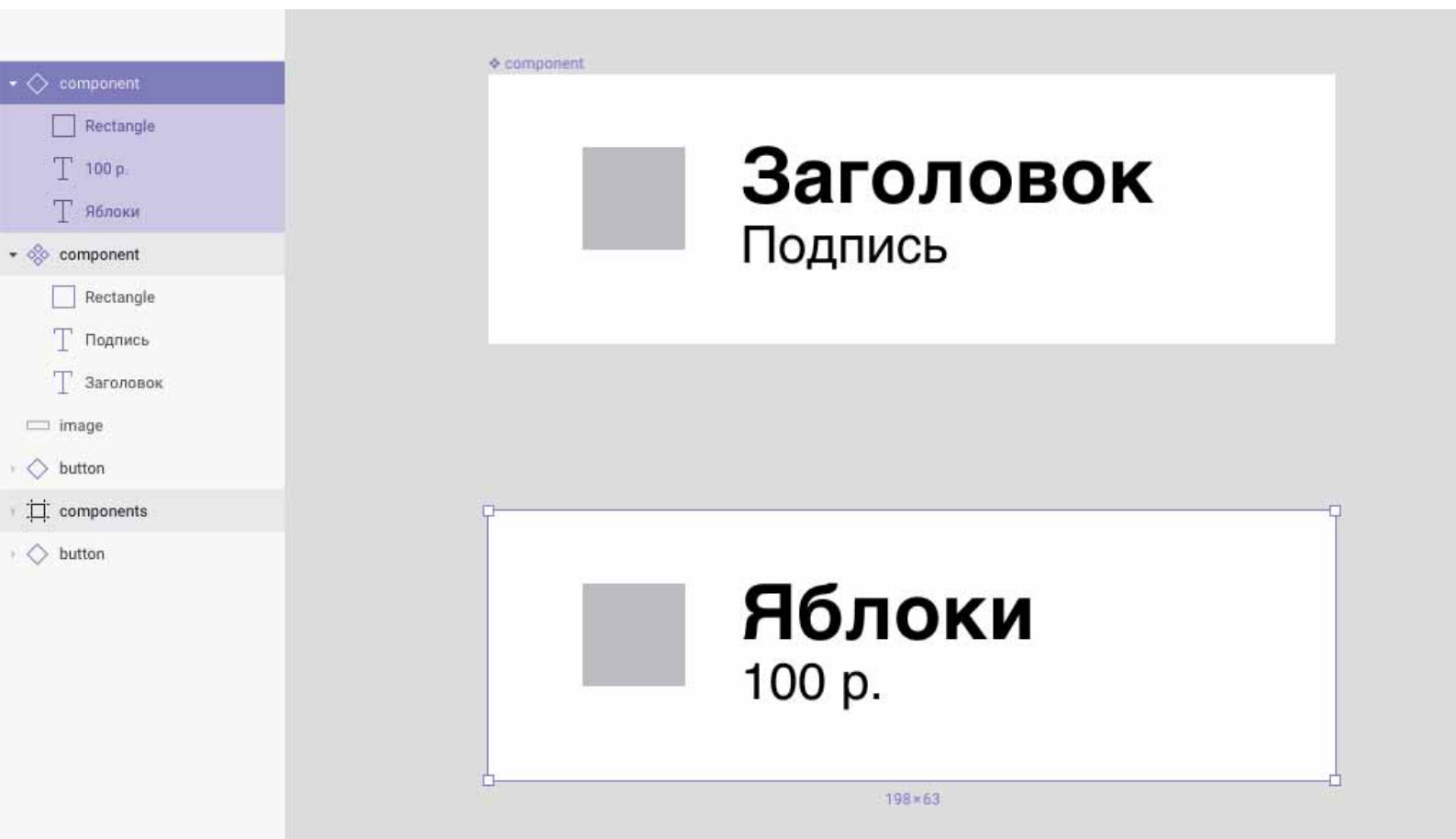
Ограничения

Весь контент, будь то текст, растровые изображения или стили можно перезаписывать только в соответствующих полях и только если последовательность слоёв внутри мастер-артборда понятна, а сами слои имеют читаемые названия. Это решение не визуально и усложняет адаптацию дизайнеров к использованию символов. Нельзя за один клик обнулить контент экземпляра.

Возможности

Однако, в нём есть и плюс: можно выделить неограниченное количество родственных экземпляров и перезаписать им один и тот же общий оверрайд одним действием.

Фигма: экземпляр компонента можно раскрыть как группу и перезаписывать непосредственно в рабочей области.



Ограничения

- Не удастся перезаписать одним текстом сразу несколько экземпляров.
- В структуру экземпляра нельзя внедрить посторонние слои, например, растровые картинки, однако можно перезаписывать заливку.

Возможности

Реализация даёт широкие возможности по работе со стилем и видимостью элементов компонента. Если в компоненте есть другой компонент, например, иконка, его можно переключать на другой в правой части интерфейса в поле **Instance**. Клавишей **Delete** можно скрывать элементы в экземплярах, что очень удобно.

Также можно обнулить оверрайды командой **Reset Instance**, которая есть в контекстном меню. Она приведёт экземпляр в исходное состояние. Через поиск: **Cmd + /, rese**.

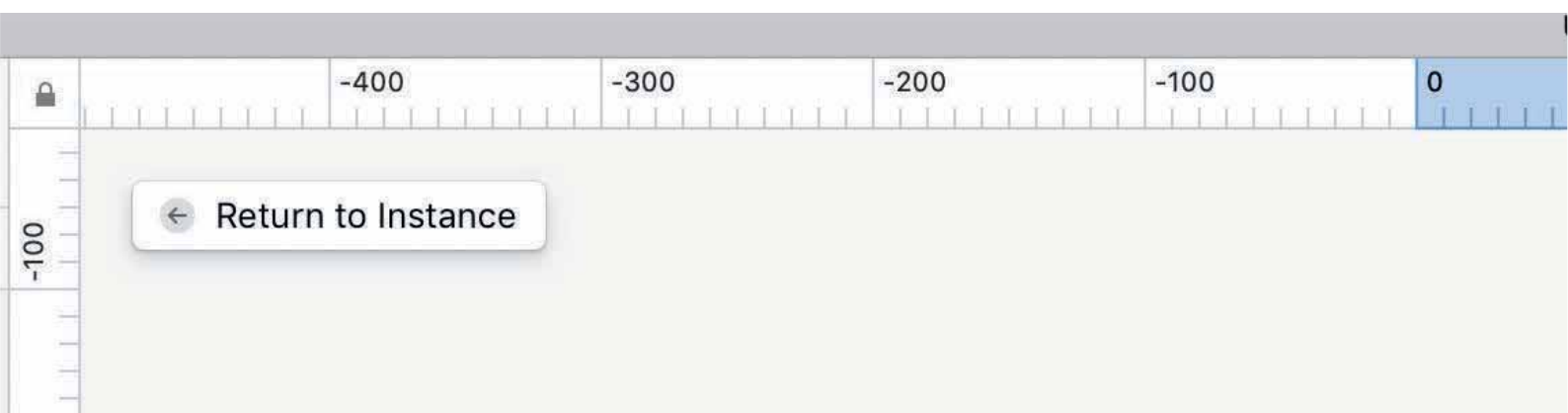
Хорошая практика: Не использовать мастера в качестве экземпляров. Если решаем сделать объект компонентом, оттаскиваем его из макета, а лучше отправляем в страницу «Компоненты». Иначе он может смешаться с экземплярами и потеряться в рядовом экране.

В Скетче такой проблемы нет, поскольку мастер-артборд и экземпляр не спутаешь: экземпляр нельзя редактировать напрямую.

Переход к мастеру

Как в Скетче, так и в Фигме, находясь в экземпляре, мы можем перейти к мастеру.

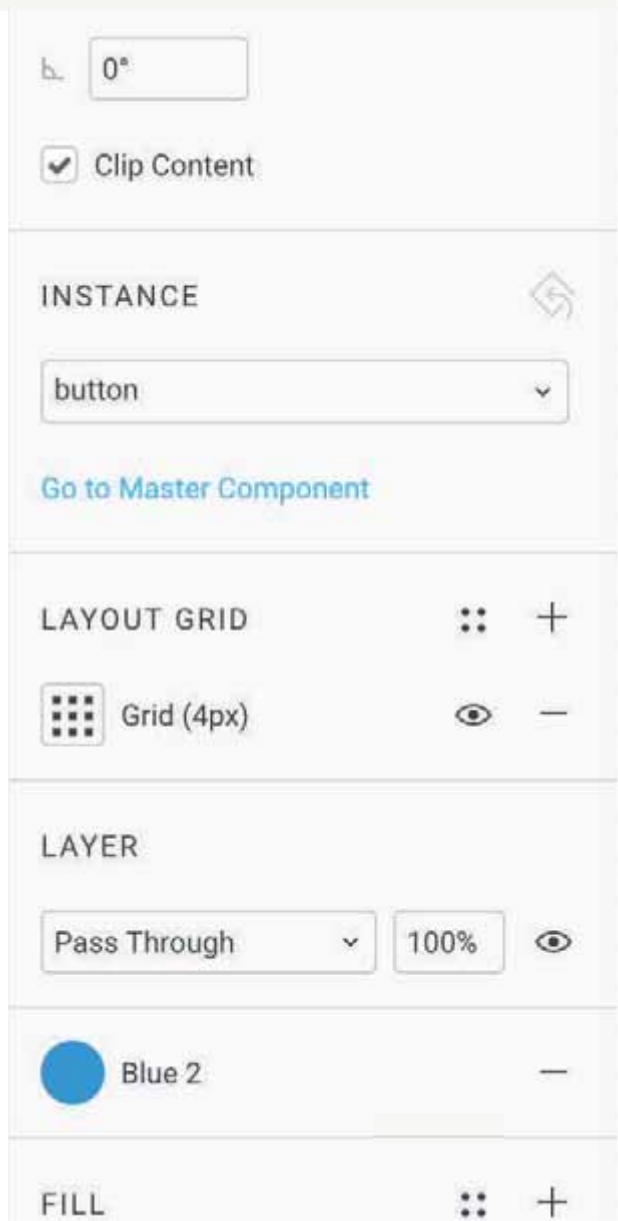
Скетч: выделяем экземпляр, нажимаем **Enter**. Чтобы вернуться к экземпляру после редактирования мастера, нажимаем команду **Return to Instance**, **Cmd + Esc**.



Фигма: **Enter** разворачивает группу экземпляра и не уводит на мастер. Чтобы перейти на него, нужно использовать команду **Go To Master Component** в блоке **Instance** или в контекстном меню.

Через поиск: **Cmd + /, go**

Команды возврата к экземпляру, аналогичной **Return to Instance** в Фигме нет, поэтому, перед прыжком хорошо бы запомнить место в проекте, откуда мы прыгаем к редактированию мастера.



Лучшие практики работы с компонентами

1. Любой компонент в дизайн-системе должен ложиться в сетку, желательно 4 или 8px.
2. Нежелательно, чтобы в компонентах был какой-либо осмысленный контент. Их нужно делать максимально обезличенными.
3. Не следует использовать мастер-компонент в качестве экземпляра в макете, его нужно выносить отдельно.
4. Не следует делать из компонентов всё, поскольку интерфейс будет очень жёстким. Некоторые объекты в дизайне всё-таки лучше оставлять группами.
5. В первую очередь нужно настраивать стили текста и цветов. Точно не стоит делать компоненты для хранения цветов.
6. Нужно дважды подумать, прежде чем заключать в компонент заголовок с текстом, содержащим конкретный контент. Будет ли он использоваться достаточное количество раз, чтобы это решение было оправдано, или лишь создаст когнитивную нагрузку на дизайнеров, которые будут работать с документом?

Тема организации компонентов заслуживает отдельной книги, которая называлась бы вроде «Как создать дизайн-систему».

Практика: компоненты

1. Создать поле ввода высотой 56px в состоянии пустого и заполненного. Должна быть подпись поля и введённый в него текст.
2. Создать экземпляр поля, переключать в нём состояния между пустым и заполненным.
3. При помощи ограничителей сделать так чтобы поле корректно тянулось на разной ширине. Особое внимание уделить длинным текстам в экземпляре.
4. Сделать две иконки размером 24 x 24, сделать из них компоненты.
5. Вставить экземпляр иконки в оба мастера поля.
6. В экземпляре поля переключать иконки через оверрайд.



Об авторе

Последние несколько лет я разрабатываю сложные банковские интерфейсы для веба и мобилок.

Работал в Почта Банке и Газпромбанке, где проектировал, делал дизайн интерфейсов и разрабатывал дизайн-системы.

Долгое время использовал Скетч и пришёл к мысли написать о нём учебник. Освоил Фигму и понял, что Скетч больше не нужен. Поэтому, написал учебник о Фигме.

[@okunev](#)