


UX/UI-ДИЗАЙН. ЗНАКОМСТВО С ИНСТРУМЕНТОМ FIGMA

 Модуль 1, Занятие №6

Тема и цель



Тема:

Мастер-класс
по созданию прототипа приложения
и графических элементов в Figma

Цель занятия:

Изучение основ по работе
с инструментом Figma

- 1. UI-дизайнер** – специалист, отвечающий за оформление продукта или сайта. От него зависит, насколько удобным для пользователя будет интерфейс.
- 2. UI-специалист** – специалист, разрабатывающий user flow (создание схемы взаимодействия потенциального клиента с сайтом или программой для выполнения той или иной задачи) и прототип (разработка интерактивного прототипа, который будет реагировать на действия пользователей).
- 3. UX/UI-дизайн** - проектирование любых пользовательских интерфейсов, в которых удобство использования так же важно, как и внешний вид.
- 4. UX-специалист** - специалист, который работает над созданием удобного продукта для конечного пользователя. Для такого специалиста особо важны положительные эмоции по работе с такими продуктами.
- 5. Веб-дизайн** - отрасль веб-разработки и разновидность дизайна, в задачи которой входит проектирование пользовательских веб-интерфейсов для сайтов или веб-приложений.

Подтемы



1. Для чего используется DataGrip, при работе с приложениями?
2. Какую основную функцию выполняет операция Select All Occurences?
3. Что такое редактор и какими они бывают?
4. Какой этап разработки является самым посредственным в среде PyCharm?

UX-ДИЗАЙН

UX-дизайн

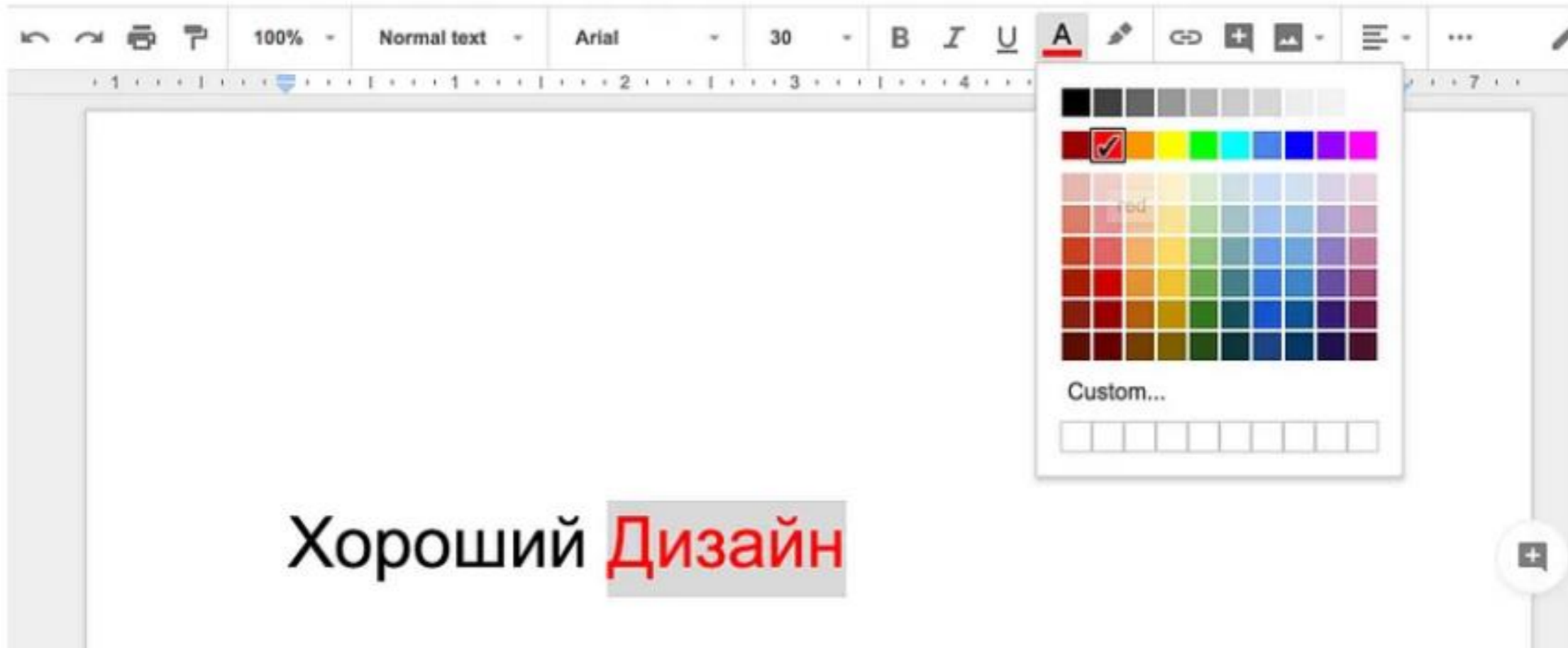


Рисунок 1 - Принцип дизайна «обратная связь»

ОТЛИЧИЕ UX ОТ UI

Отличие UX от UI



Рисунок 1 - Принцип дизайна «обратная связь»

Отличие UX от UI

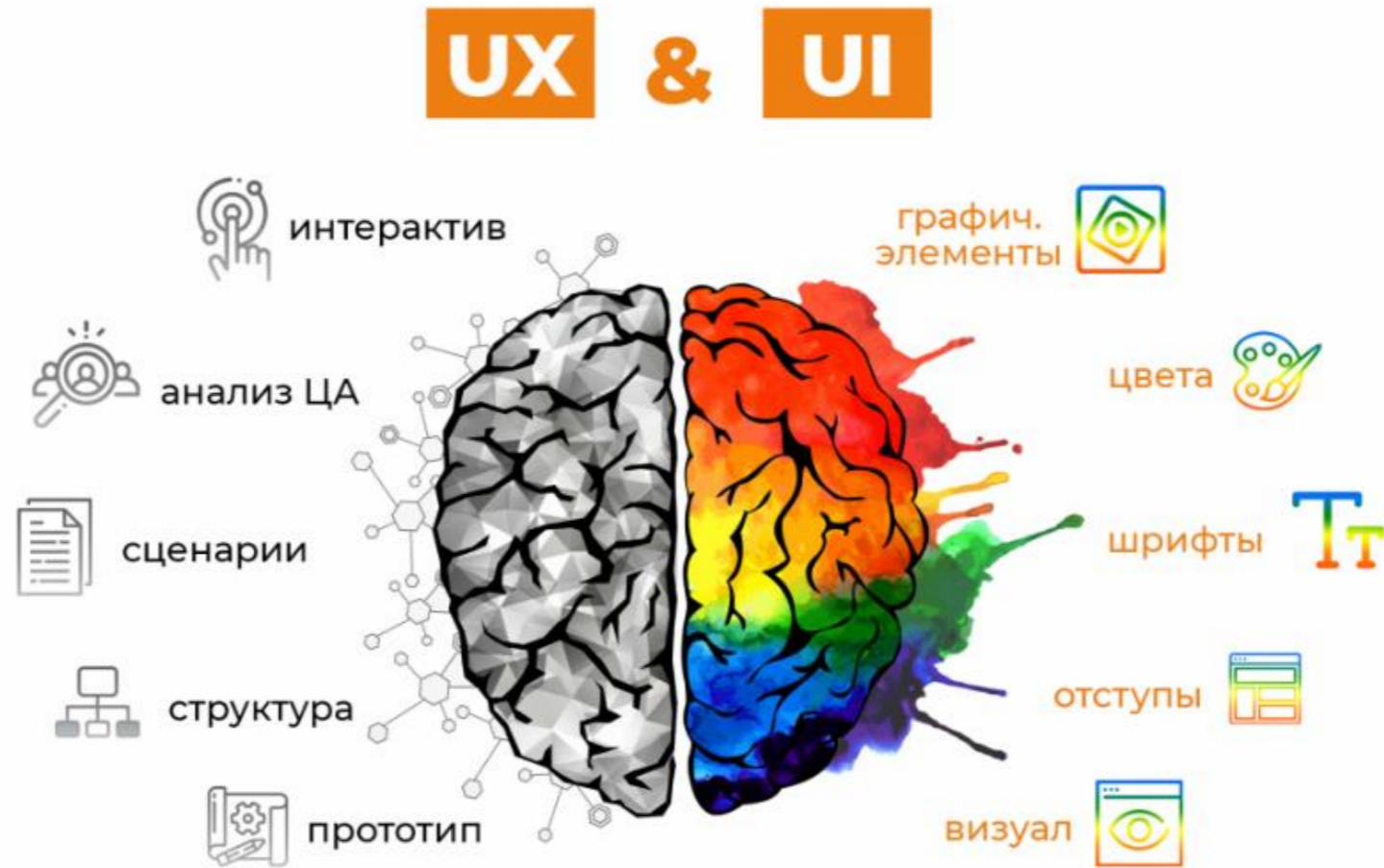


Рисунок 3 - Отличие в параметрах и компонентах UI-дизайна и UX-дизайна

Blender 3D



Пример



В данном примере используется PyQt. Это фреймворк Qt. С помощью данного фреймворка сделаны blender3d, Telegram, Ipython, Jupyter, VLC и другие.

Для начала, нужно установить PyQt. Это можно сделать при помощи следующей команды:

```
>>> pip install PyQt5
```

Пример



Также, нужно создать папку с проектом, назовем его helloApp. Откроем файл main.py и введем следующий код:

```
import sys
```

← часть кода, отвечающая за импорт **sys**

```
from PyQt5.QtGui import QApplication  
from PyQt5.QtQml import QQmlApplicationEngine
```

```
app = QApplication(sys.argv)
```

```
engine = QQmlApplicationEngine()  
engine.quit.connect(app.quit)  
engine.load('./UI/main.qml')
```

```
sys.exit(app.exec())
```

← часть кода, отвечающая за выход из файла

Пример



Затем добавим этот код в «main.qml»:

```
import QtQuick 2.15
import QtQuick.Controls 2.15
```

← импортирование
необходимых элементов

```
ApplicationWindow {
    visible: true
    width: 600
    height: 500
    title: "HelloApp"
}
```

← задание
параметров

```
Text {
    anchors.centerIn: parent
    text: "Hello, World"
    font.pixelSize: 24
}
```

← содержание текста
и размер окна

```
}
```

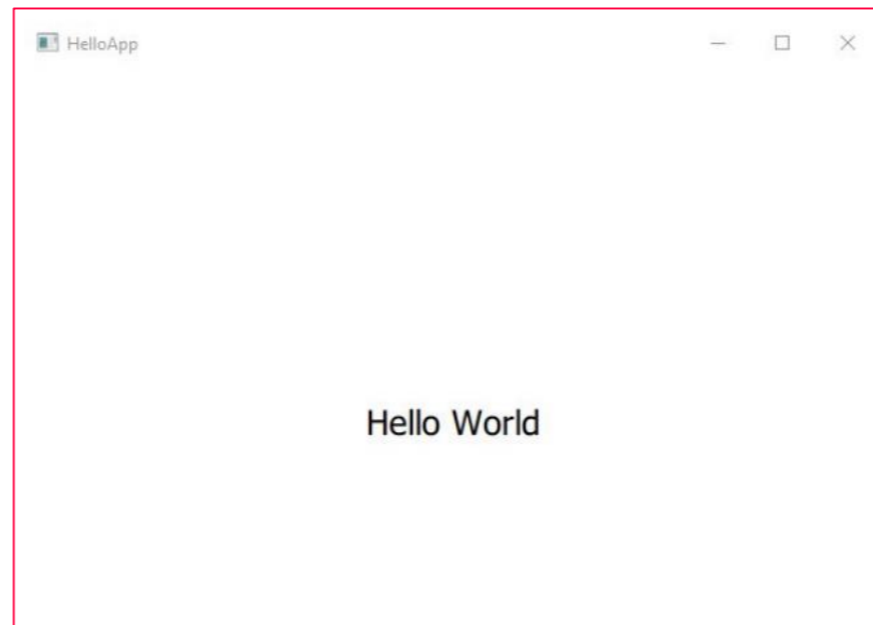
Пример



Теперь запустим приложение:

```
>>> python main.py
```

После этого мы увидим следующее окно:



Пример



Теперь немного обновим UI, добавив фоновое изображение и время:

```
import QtQuick 2.15
import QtQuick.Controls 2.15
```

```
ApplicationWindow {
    visible: true
    width: 400
    height: 600
    title: "HelloApp"
```

задание параметров
изображения



```
Rectangle {
    anchors.fill: parent
```

```
Image {
    sourceSize.width: parent.width
    sourceSize.height: parent.height
    source: "../images/playas.jpg"
    fillMode: Image.PreserveAspectCrop
```

```
}
```

```
Rectangle {
```

```
    anchors.fill: parent
```

```
    color: "transparent"
```

задание цвета
из палитры

```
    Text {
```

```
        text: "16:38:33"
```

содержимое текста

```
        font.pixelSize: 24
```

```
        color: "white"
```

цвет текста

```
    }
```

```
}
```

```
}
```


Пример



После запуска данного листинга, мы увидим следующее:





Также, существует модуль **gmtime**, который использует структуру со временем, а **strftime** дает возможность преобразовать ее в строку. Импортируем их:

```
import sys
from time import strftime, gmtime
```

Теперь можно получить строку с текущим временем:

```
curr_time = strftime("%H:%M:%S", gmtime())
```

Строка «%H:%M:%S» означает, что мы получим время в 24-часовом формате, с часами, минутами и секундами.



Подробнее о strftime можно
ознакомиться по ссылке:





Теперь давайте создадим **property** в **qml** файле, для хранения времени. Назовем его *currTime*:

```
property string currTime: "00:00:00"
```

Также заменим текст переменной:

```
Text {  
    ...  
    text: currTime // used to be; text: "16:38:33"  
    font.pixelSize: 48 ← задание размера текста  
    color: "white" ← задание цвета текста  
}
```



У нас уже есть свойства для строки со временем **curr_time**, теперь можно создать свойство **backend** типа **QObject** в файле **main.qml**:

```
property string currTime: "00:00:00"  
property QObject backend
```

И передадим данные из Python в qml:

```
engine.load('./UI/main.qml')  
back_end = Backend()  
engine.rootObjects()[0].setProperty('backend', back_end)
```

ИНСТРУМЕНТ FIGMA

Инструмент Figma

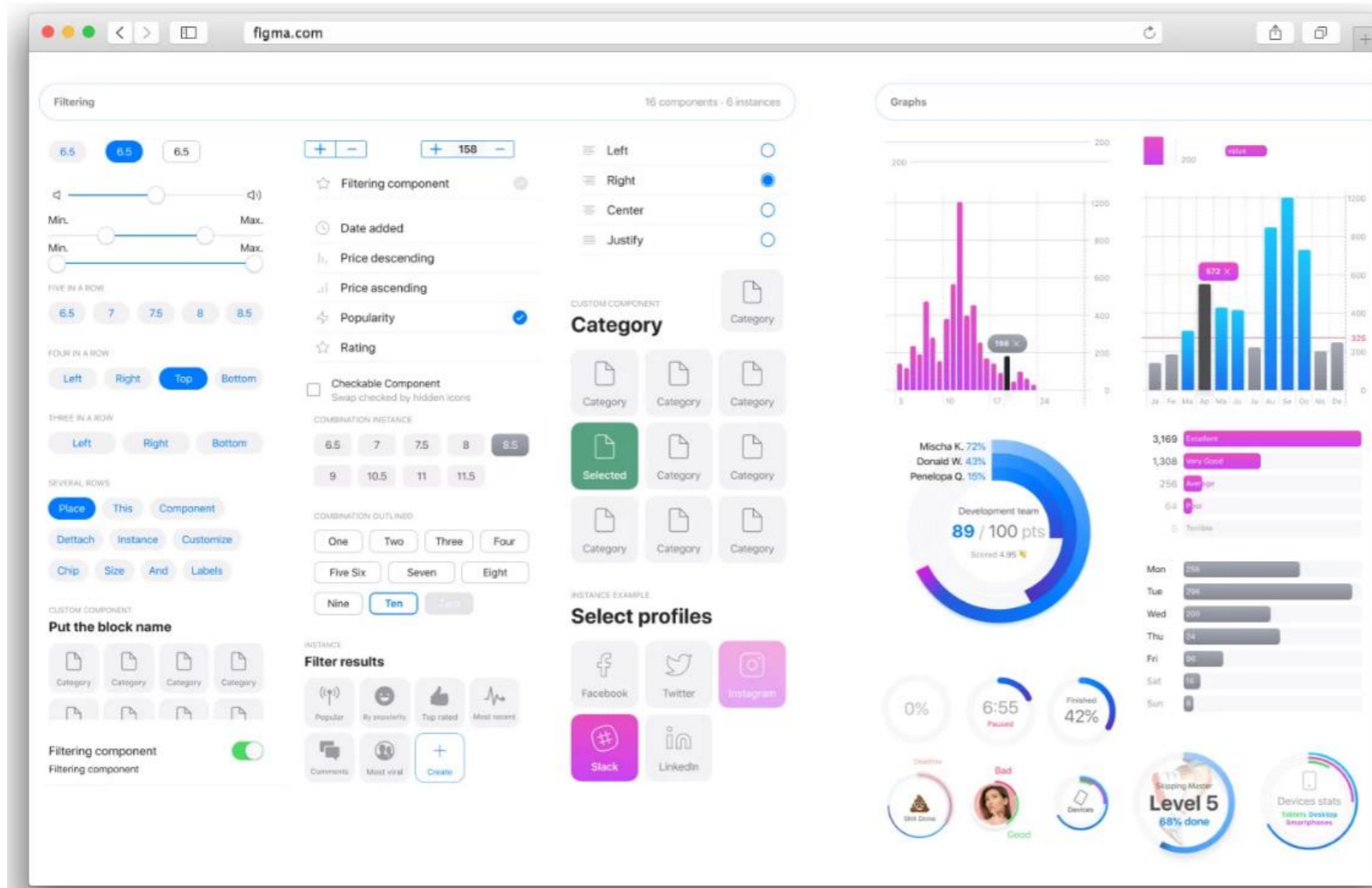


Рисунок 4 - Интерфейс инструмента Figma

ИНСТРУМЕНТ Figma

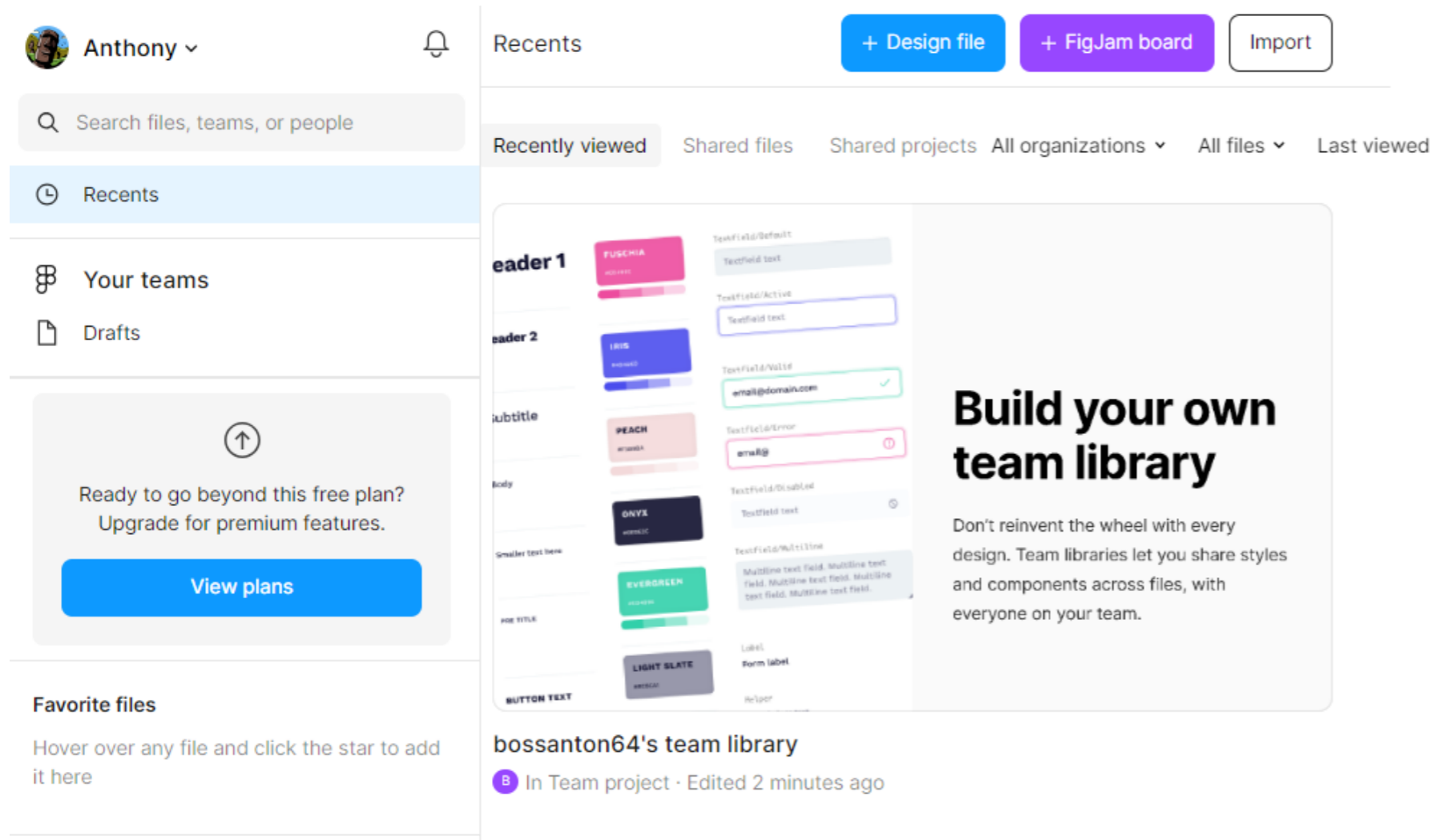


Рисунок 5 –
Стартовое меню Figma

Инструмент Figma

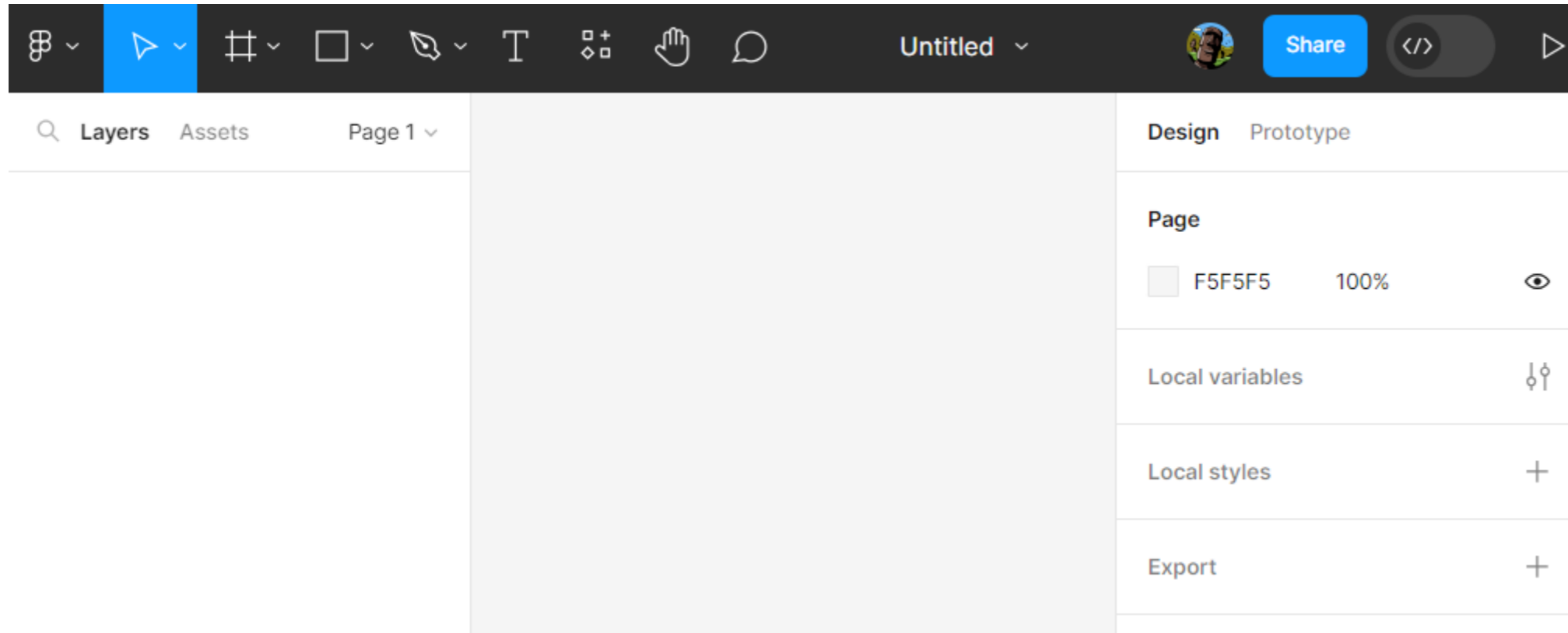


Рисунок 6 - Рабочее пространство Figma

Инструмент Figma

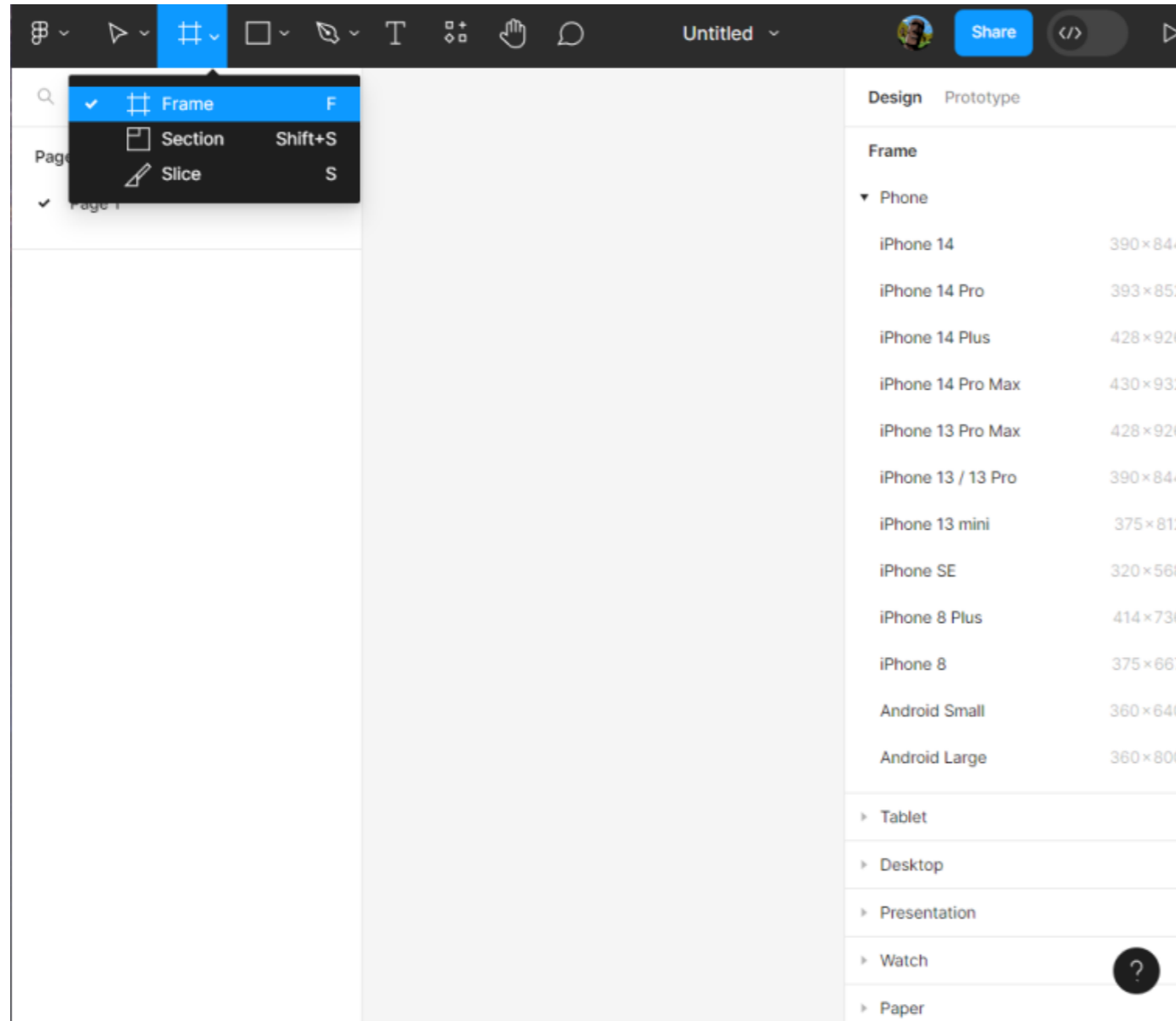


Рисунок 7 –
Добавление областей для устройств

Инструмент Figma

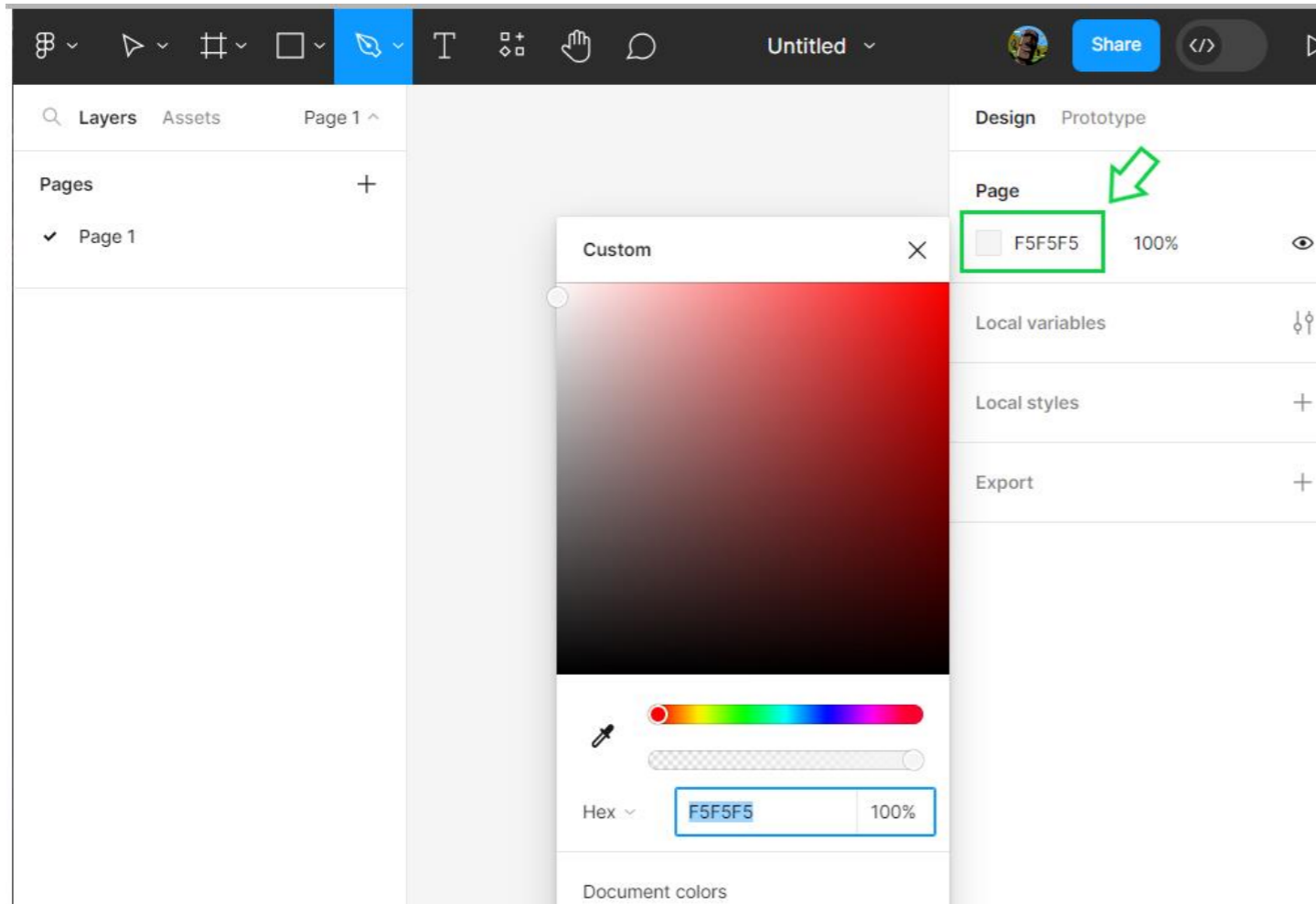


Рисунок 8 - Выбор цвета окна

Инструмент Figma

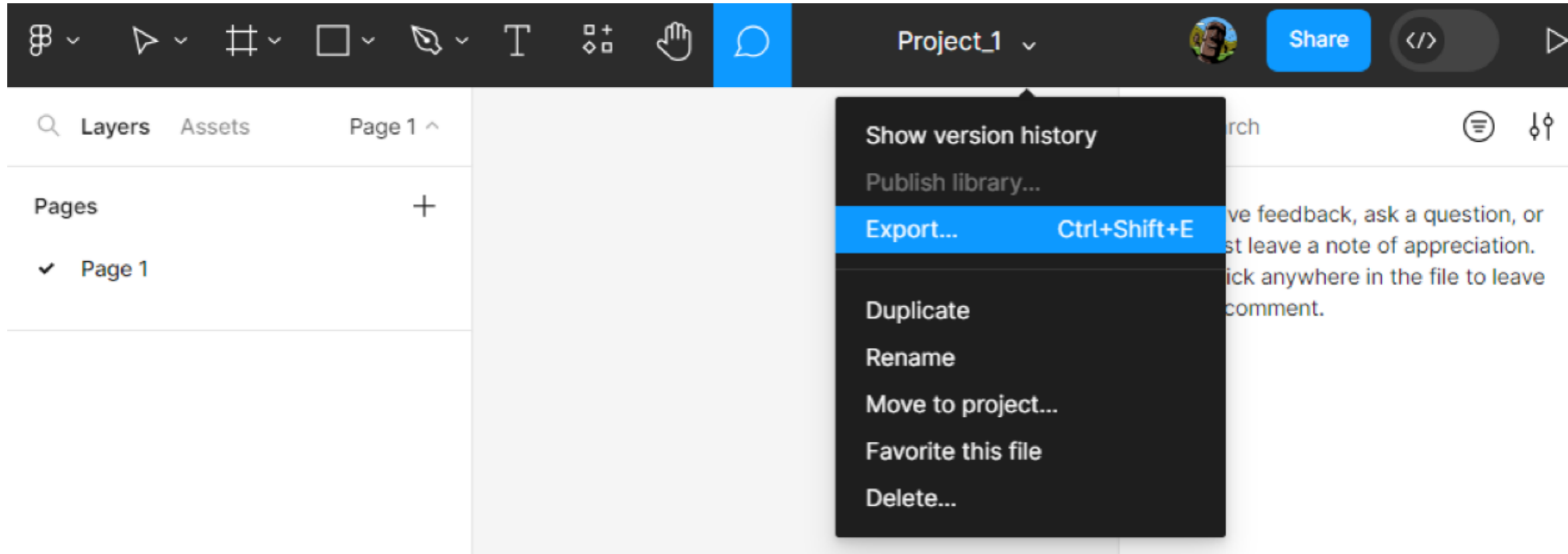


Рисунок 9 - Функция экспортирования проектов

Инструмент Figma

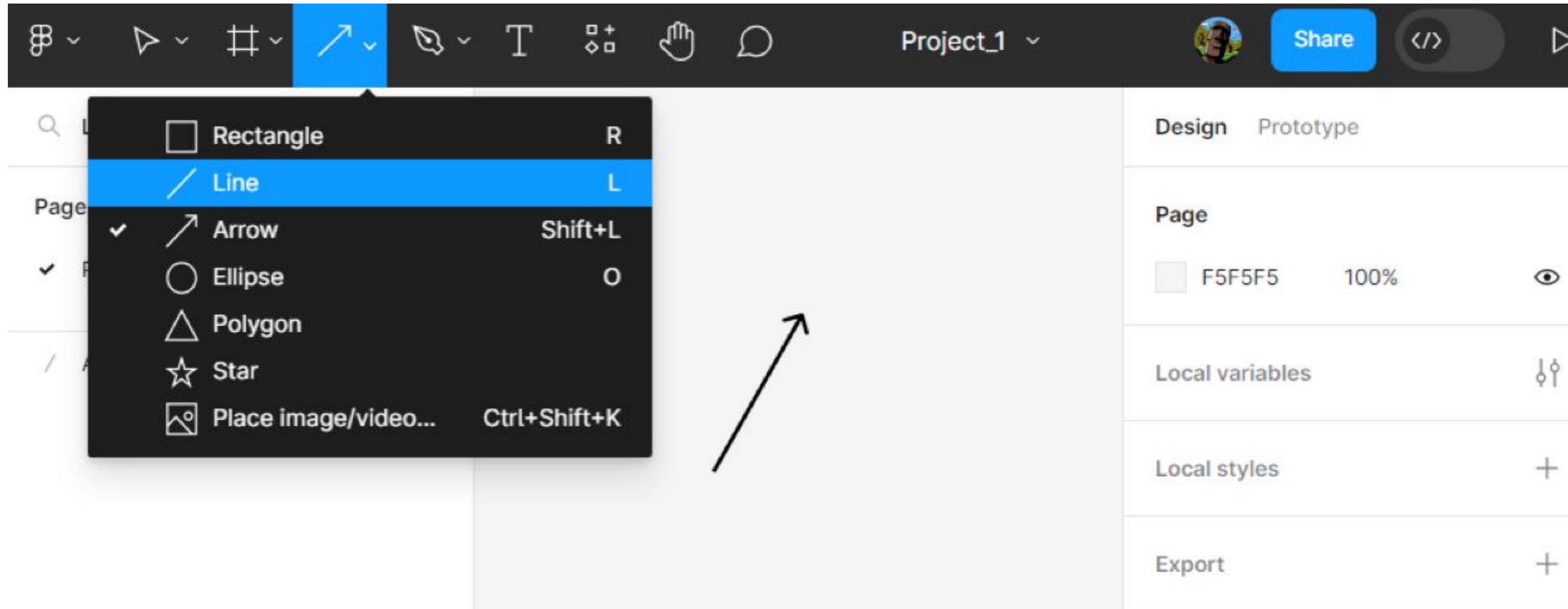


Рисунок 10 - Функция добавления 2D-фигур

Инструмент Figma

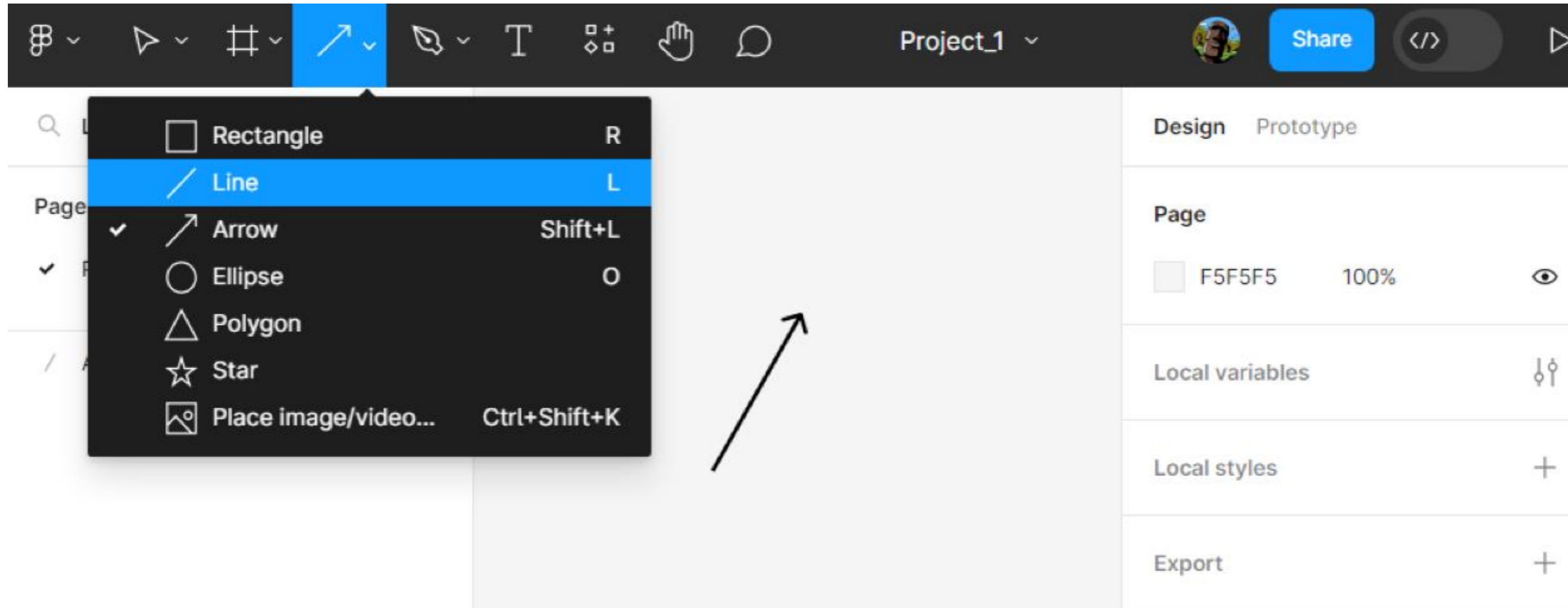


Рисунок 11 - Выделение области в проекте

Инструмент Figma

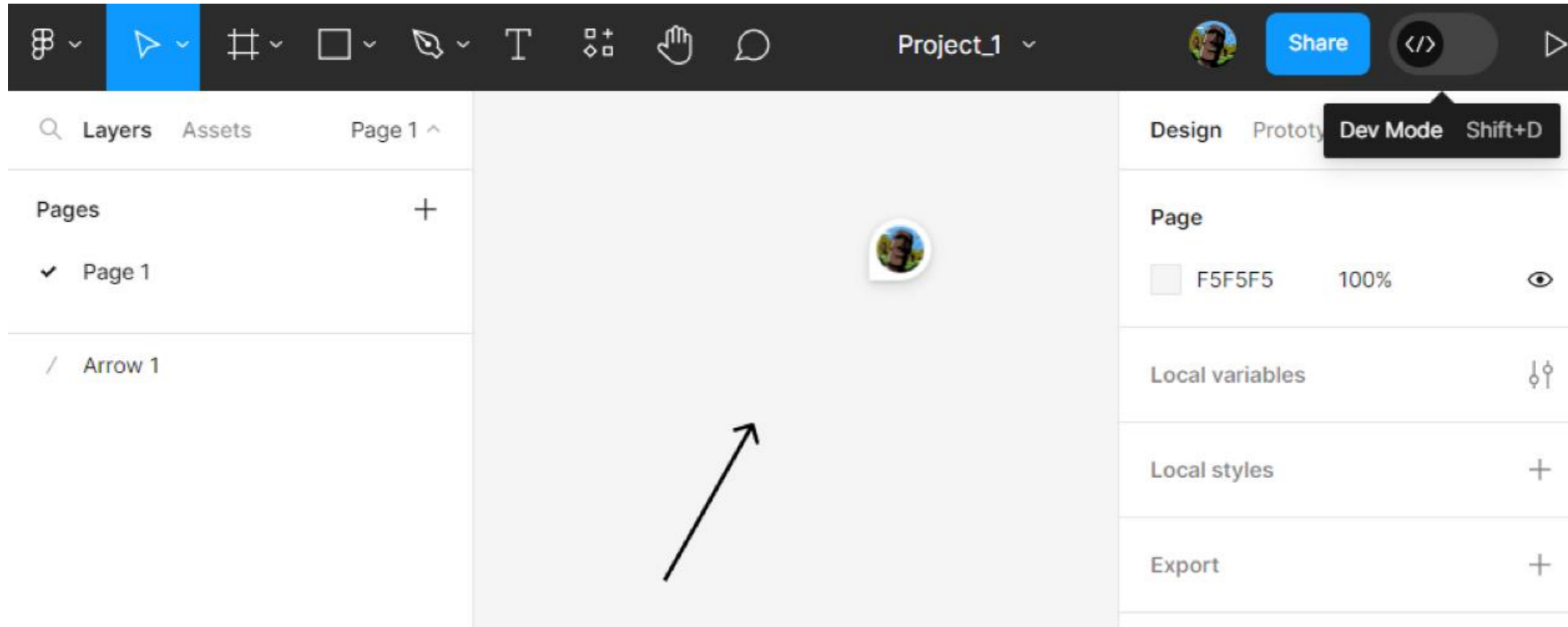


Рисунок 12 - Функция перехода в Dev Mode

Инструмент Figma

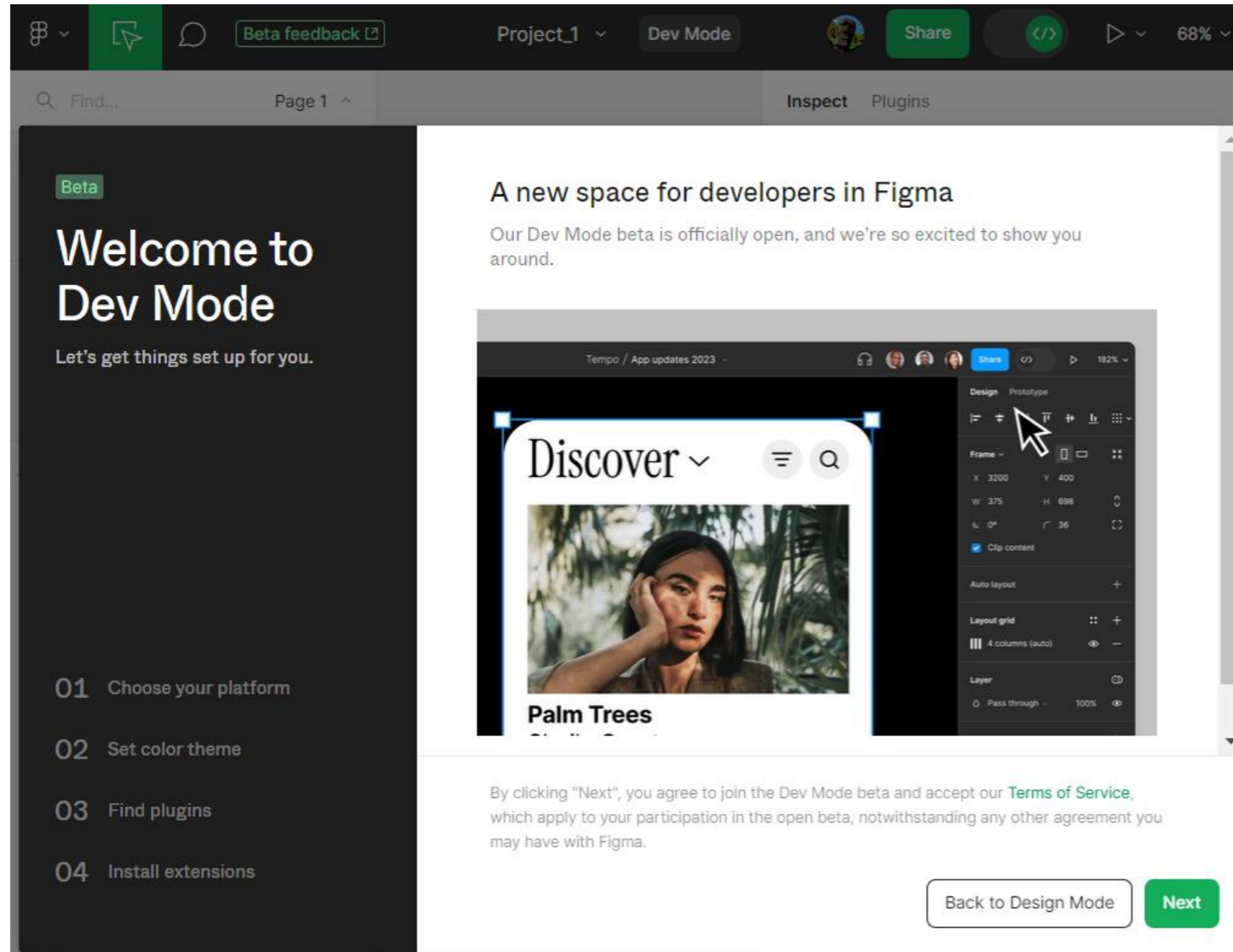


Рисунок 13 - Меню перехода в Dev Mode

Инструмент Figma



После нажатия на кнопку 'Next' нужно выбрать операционную систему, с которой будет произведена работа

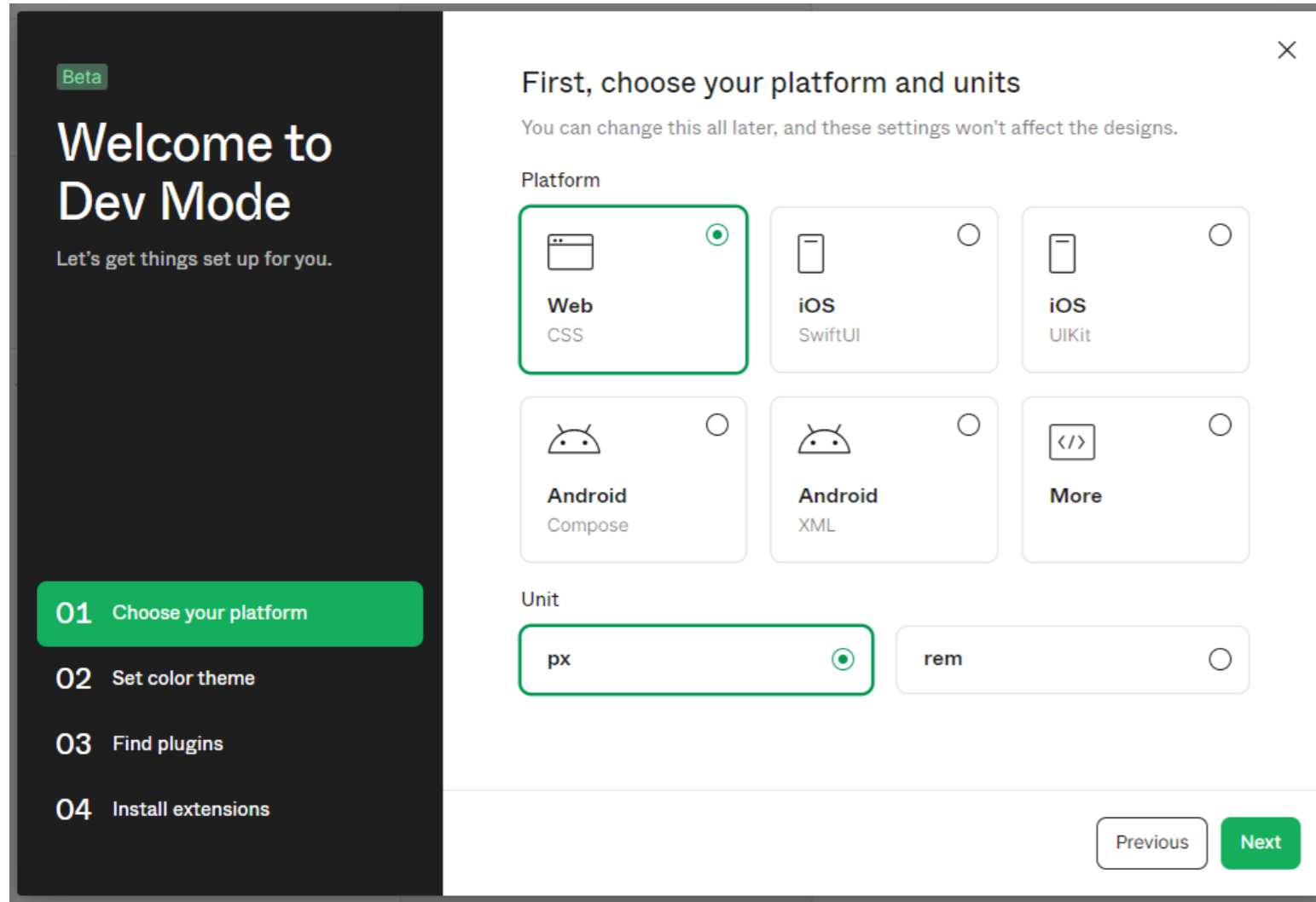


Рисунок 14 - Окно выбора операционной системы

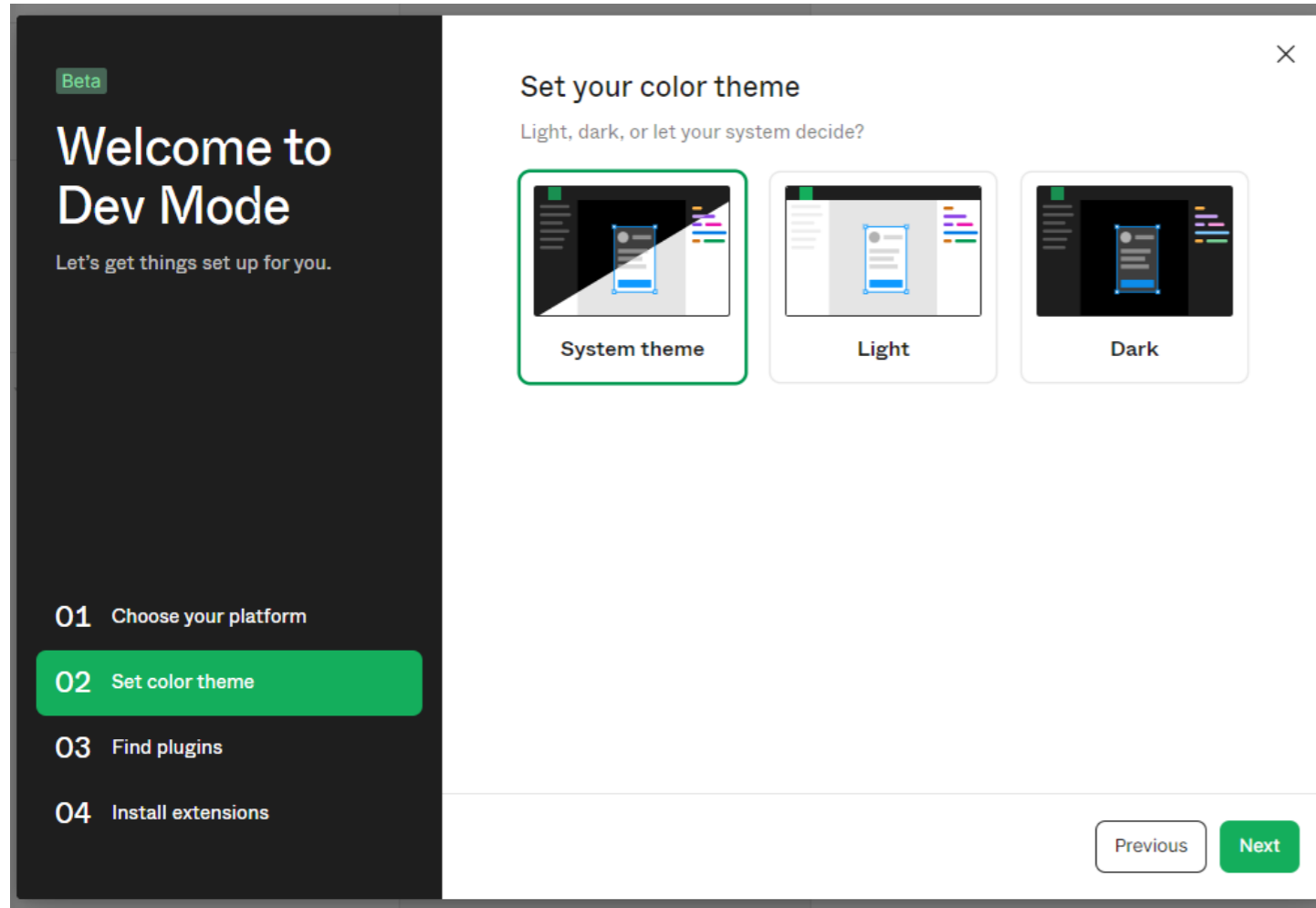


Рисунок 15 - Окно выбора цвета темы

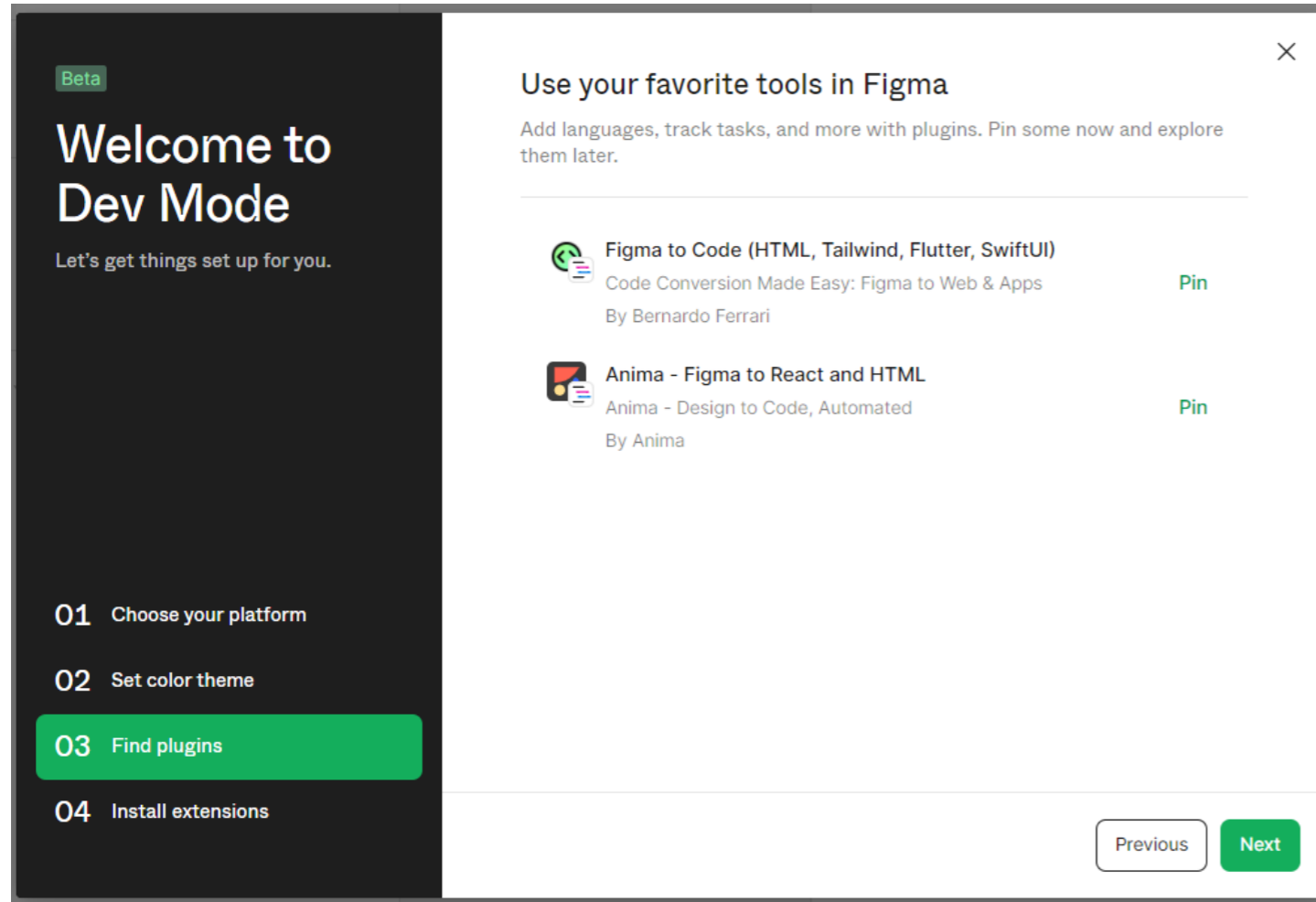


Рисунок 16 - Окно выбора подсистемы

Инструмент Figma

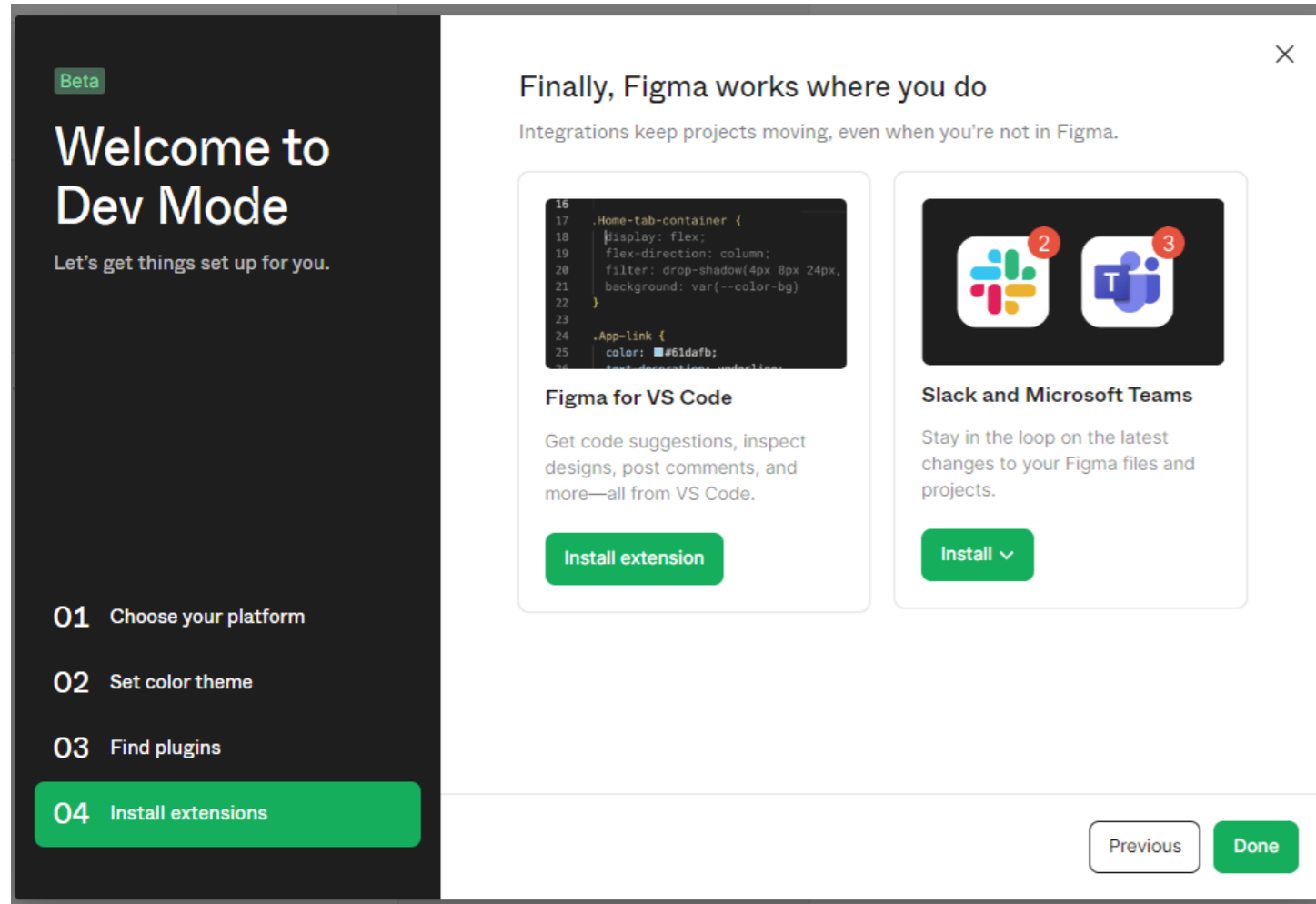


Рисунок 17 - Окно выбора среды

Инструмент Figma

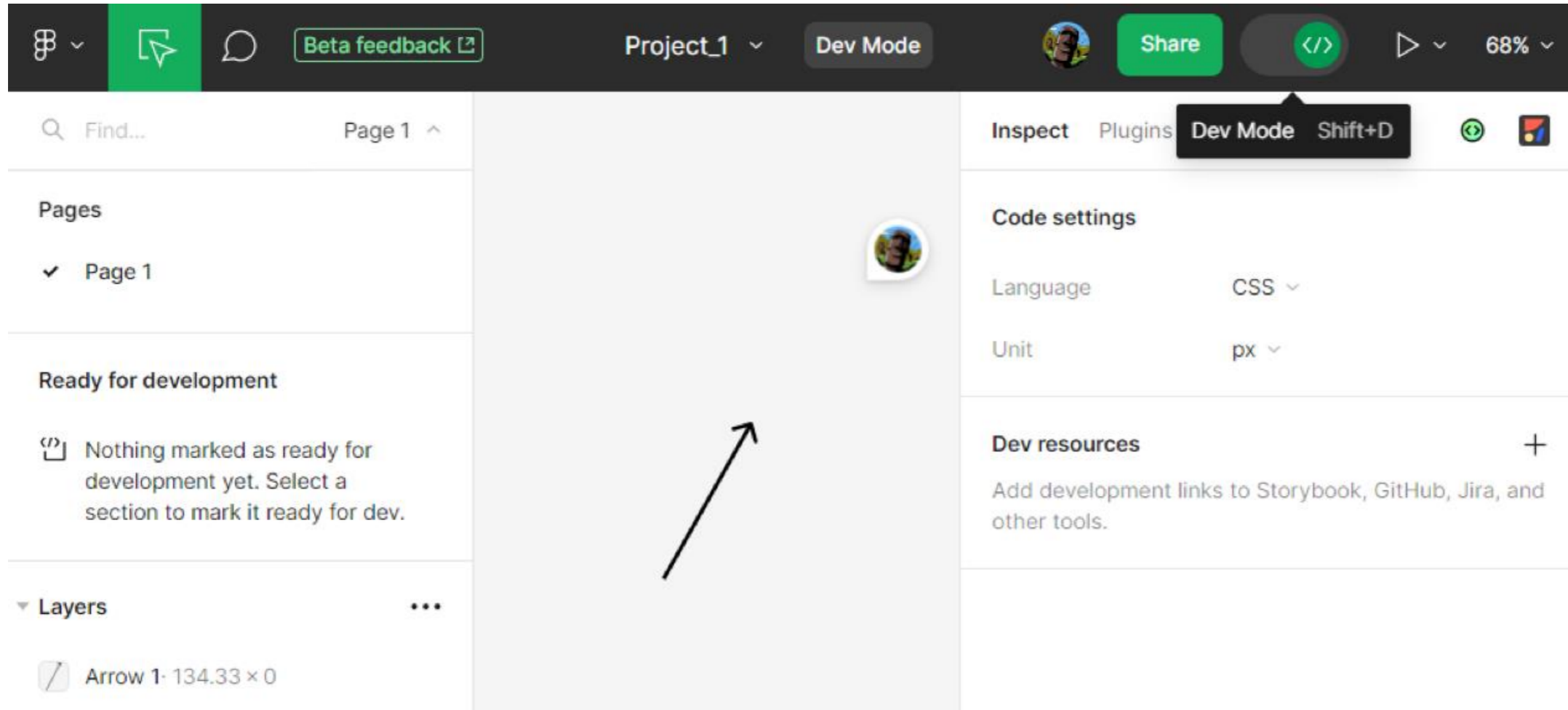


Рисунок 12 - Функция перехода в Dev Mode

Инструмент Figma

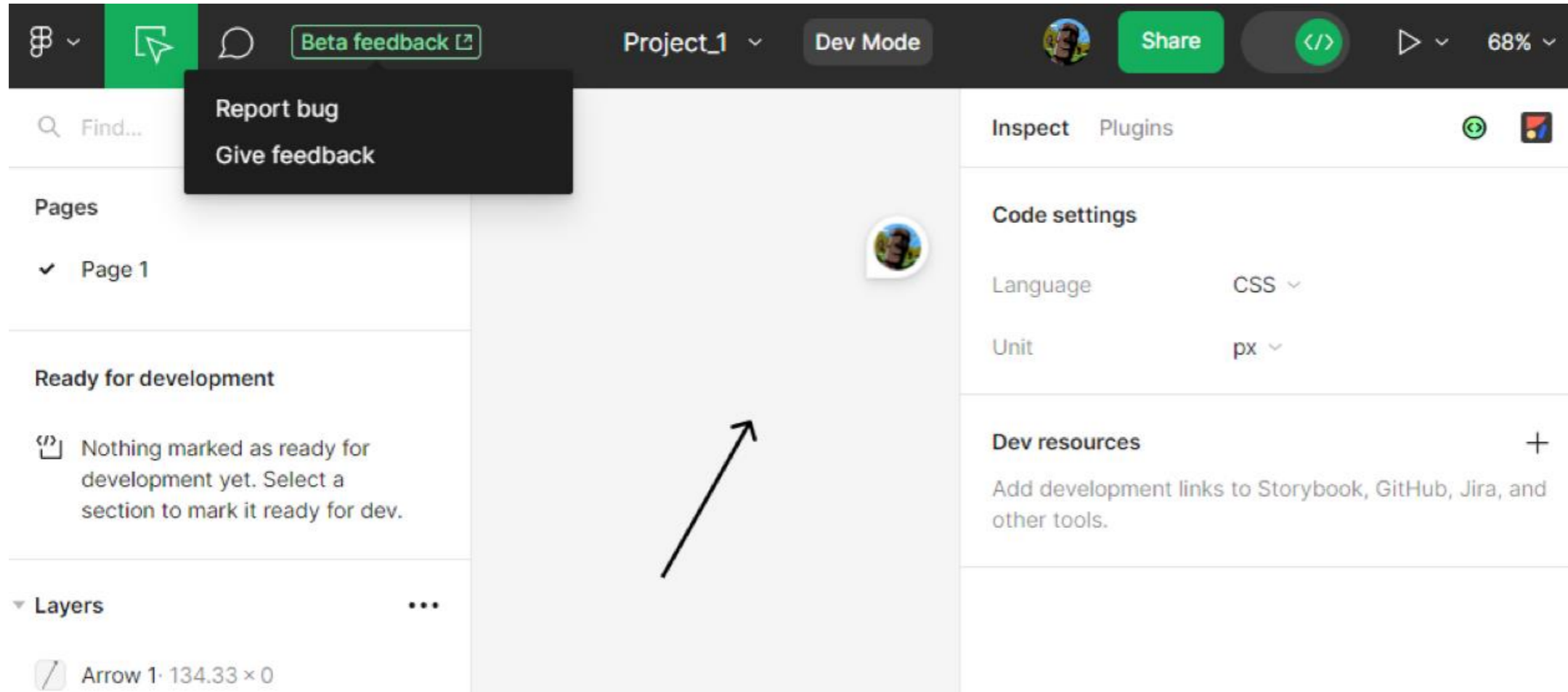


Рисунок 19 - Вид кнопки обратной связи

Инструмент Figma

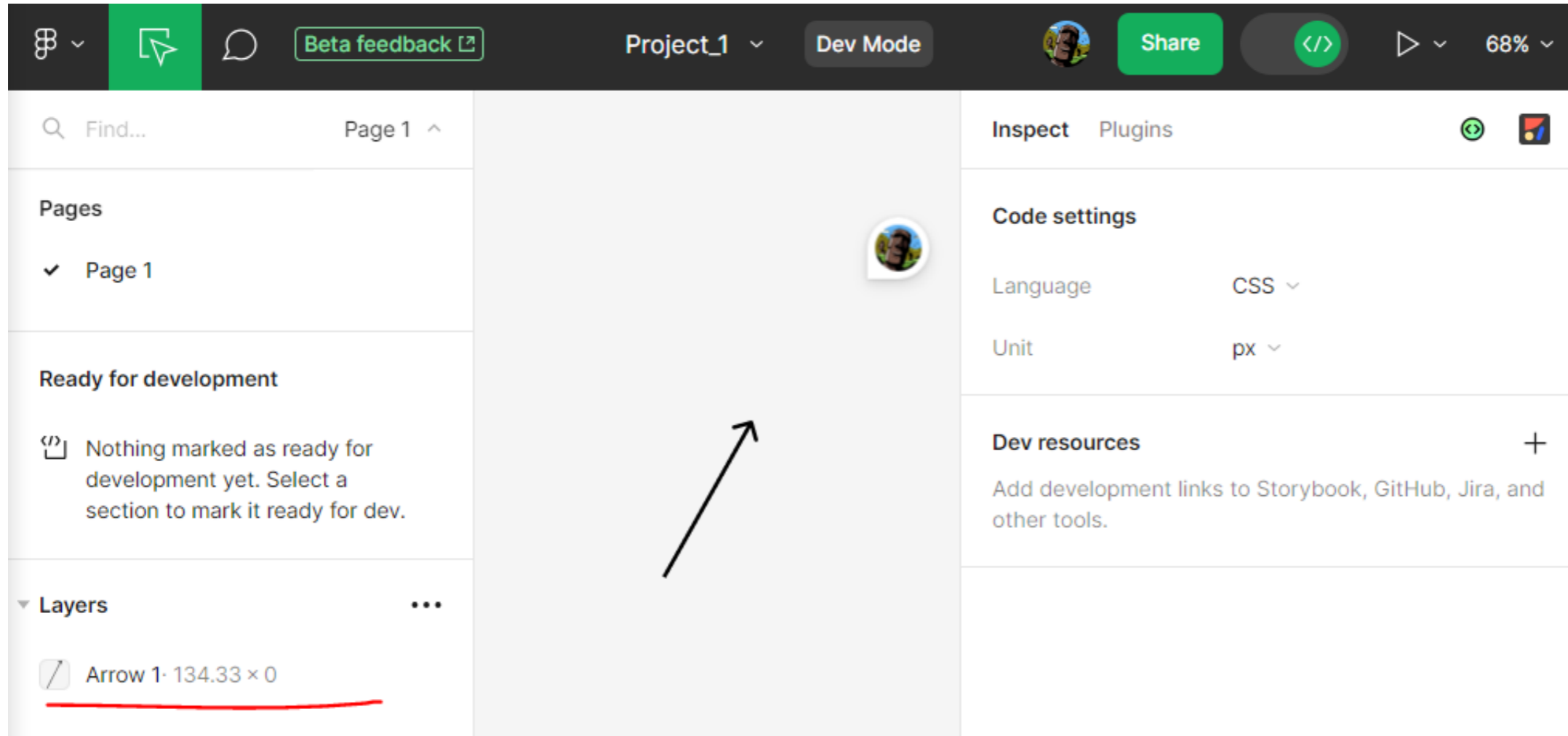


Рисунок 20 - Отображение объекта в поле 'Layers'

Инструмент Figma

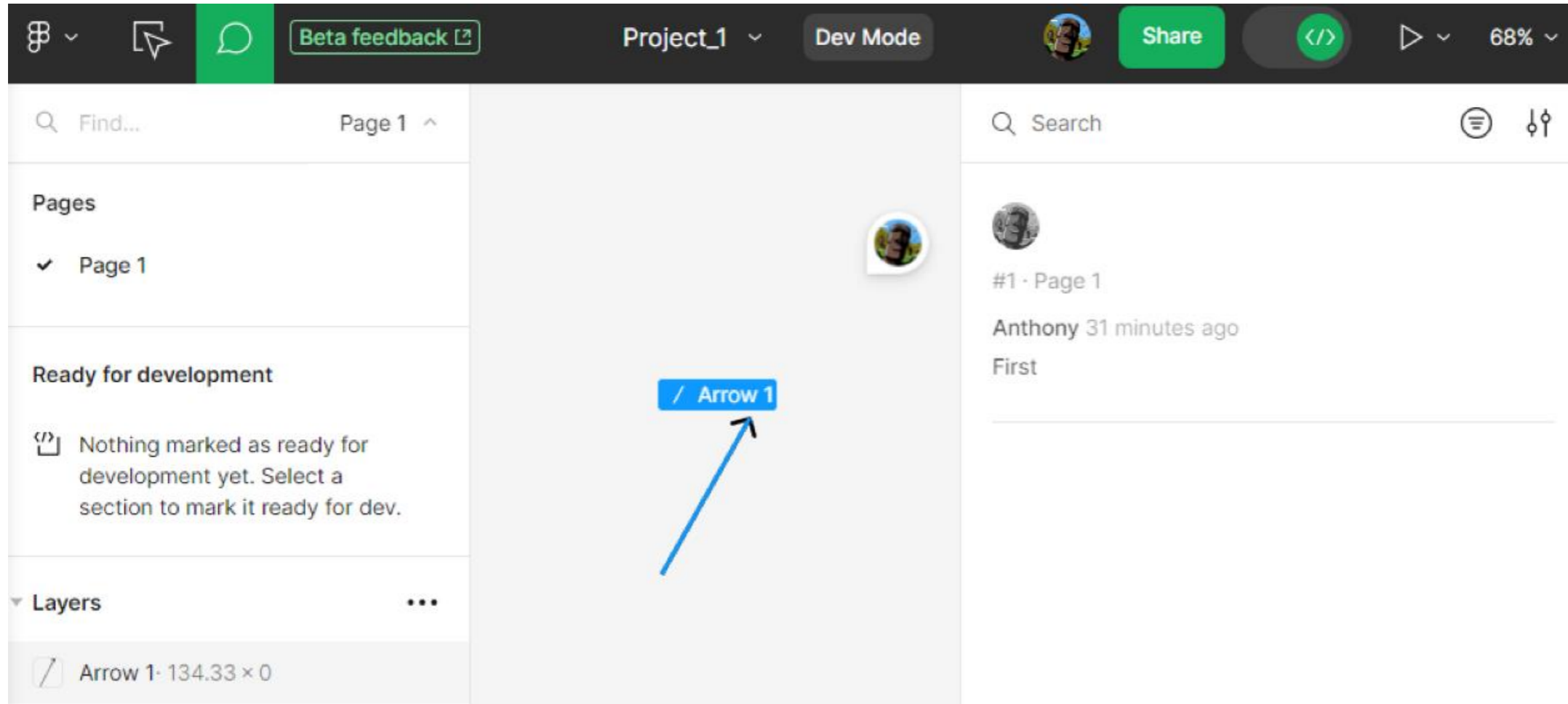


Рисунок 21 - Выбор объекта в Figma

Инструмент Figma

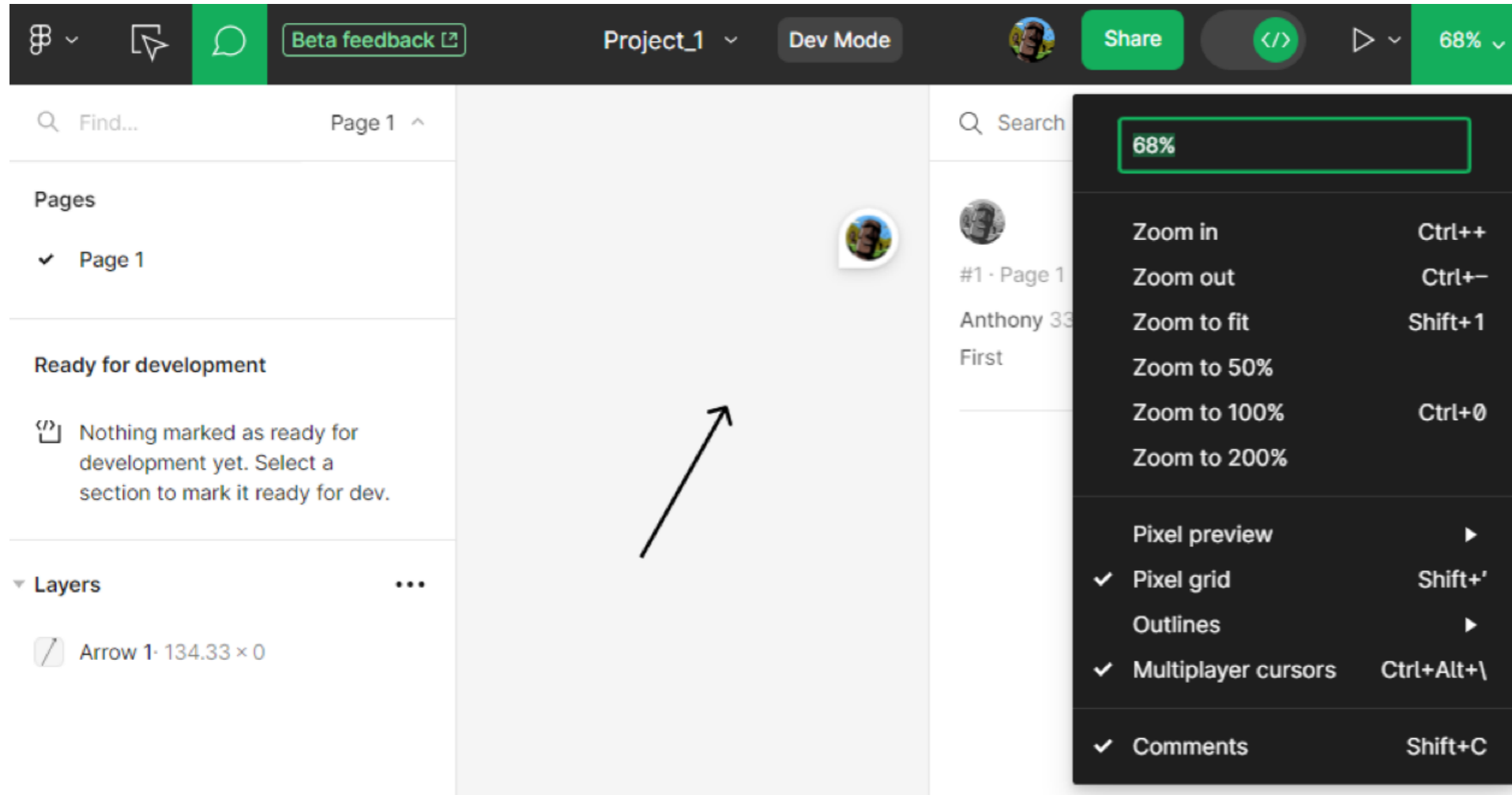


Рисунок 22 - Вид кнопки для приближения/отдаления

Инструмент Figma

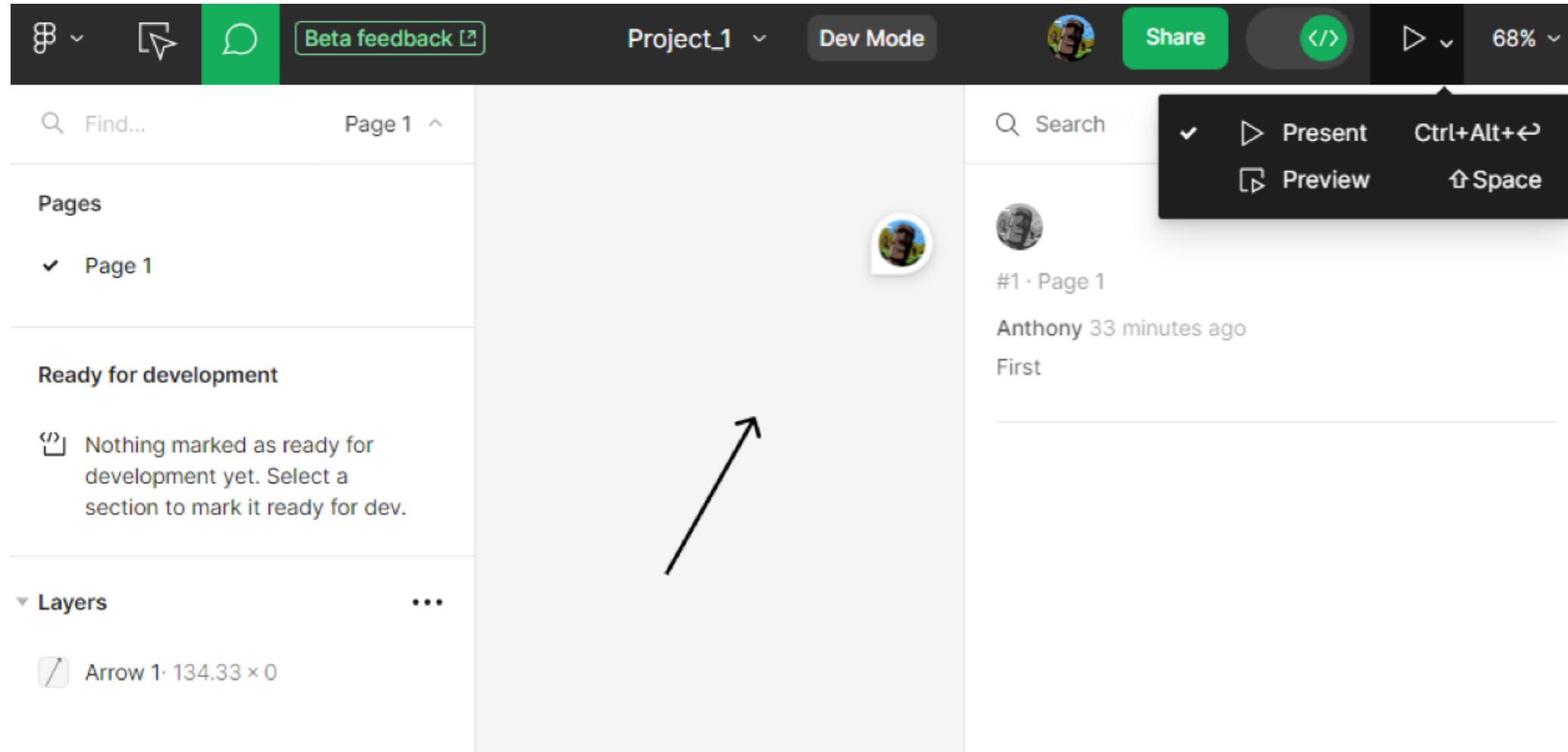


Рисунок 33 - Вид кнопки запуска

Инструмент Figma

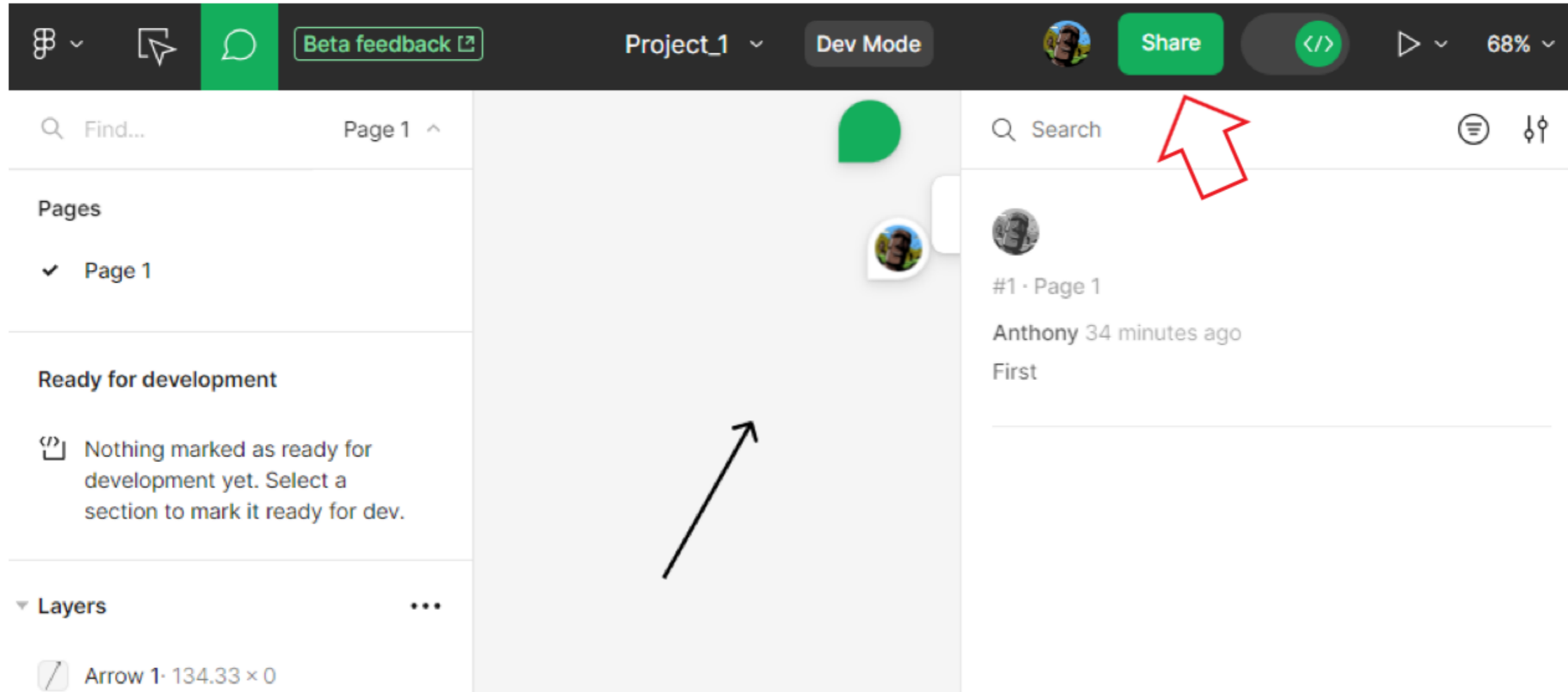


Рисунок 34 - Вид кнопки 'Share'

Инструмент Figma

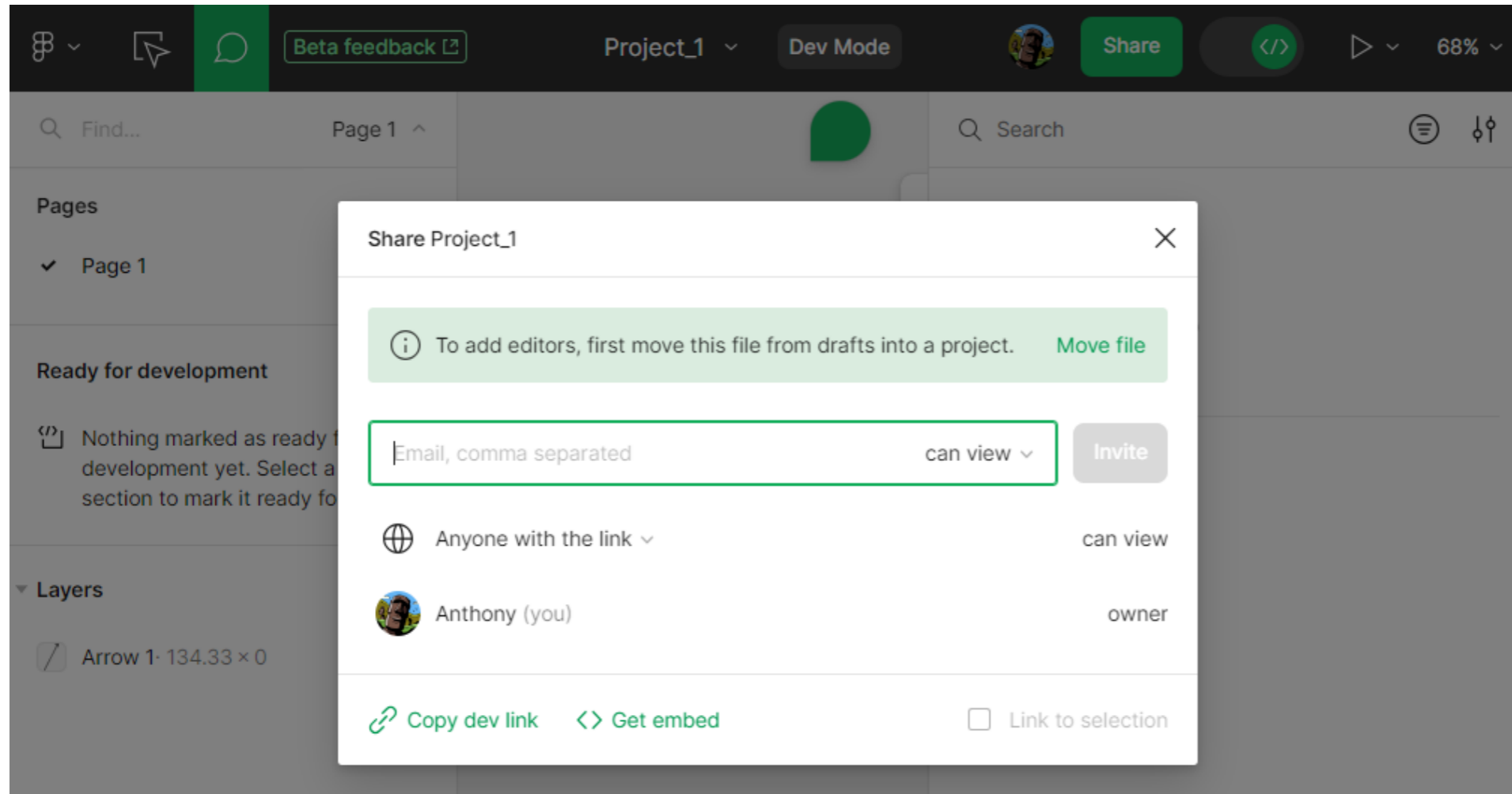


Рисунок 35 - Меню для отправки проекта

Инструмент Figma

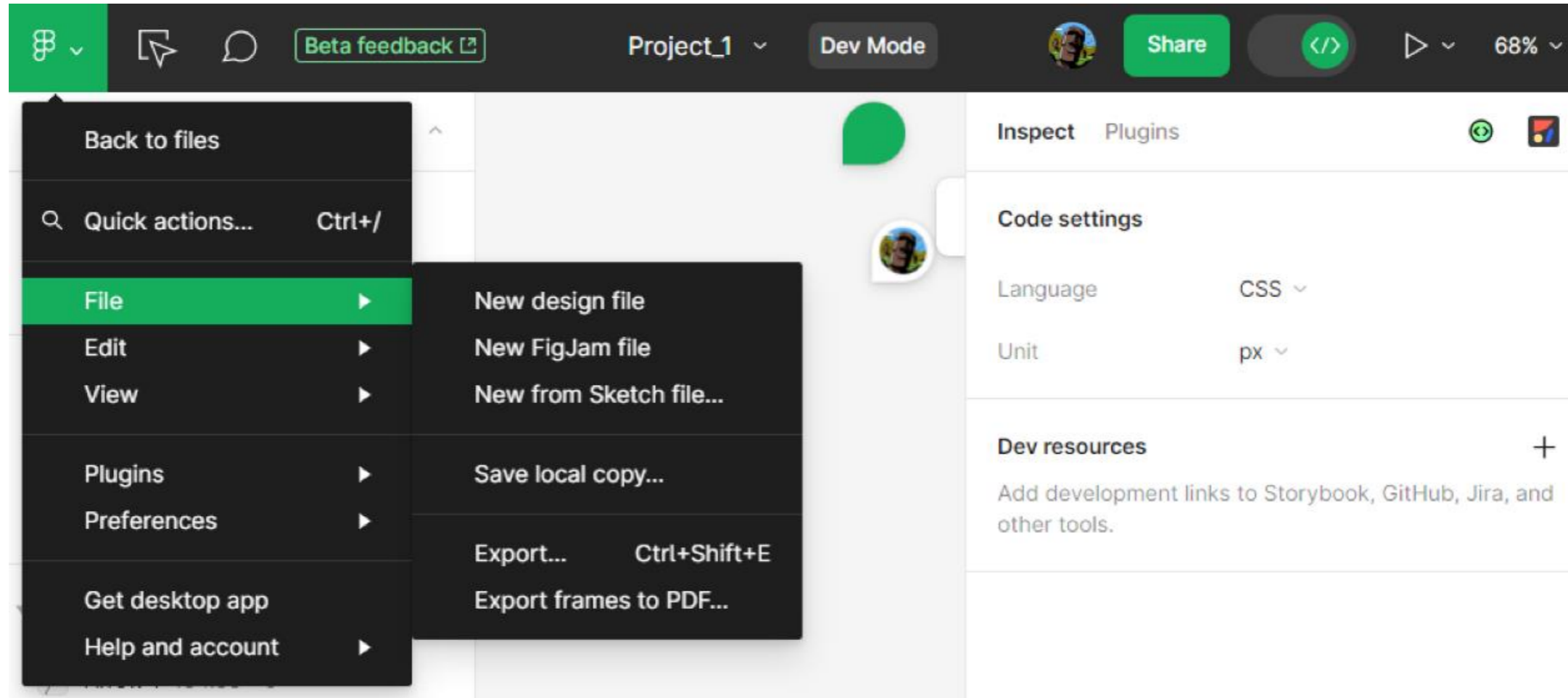
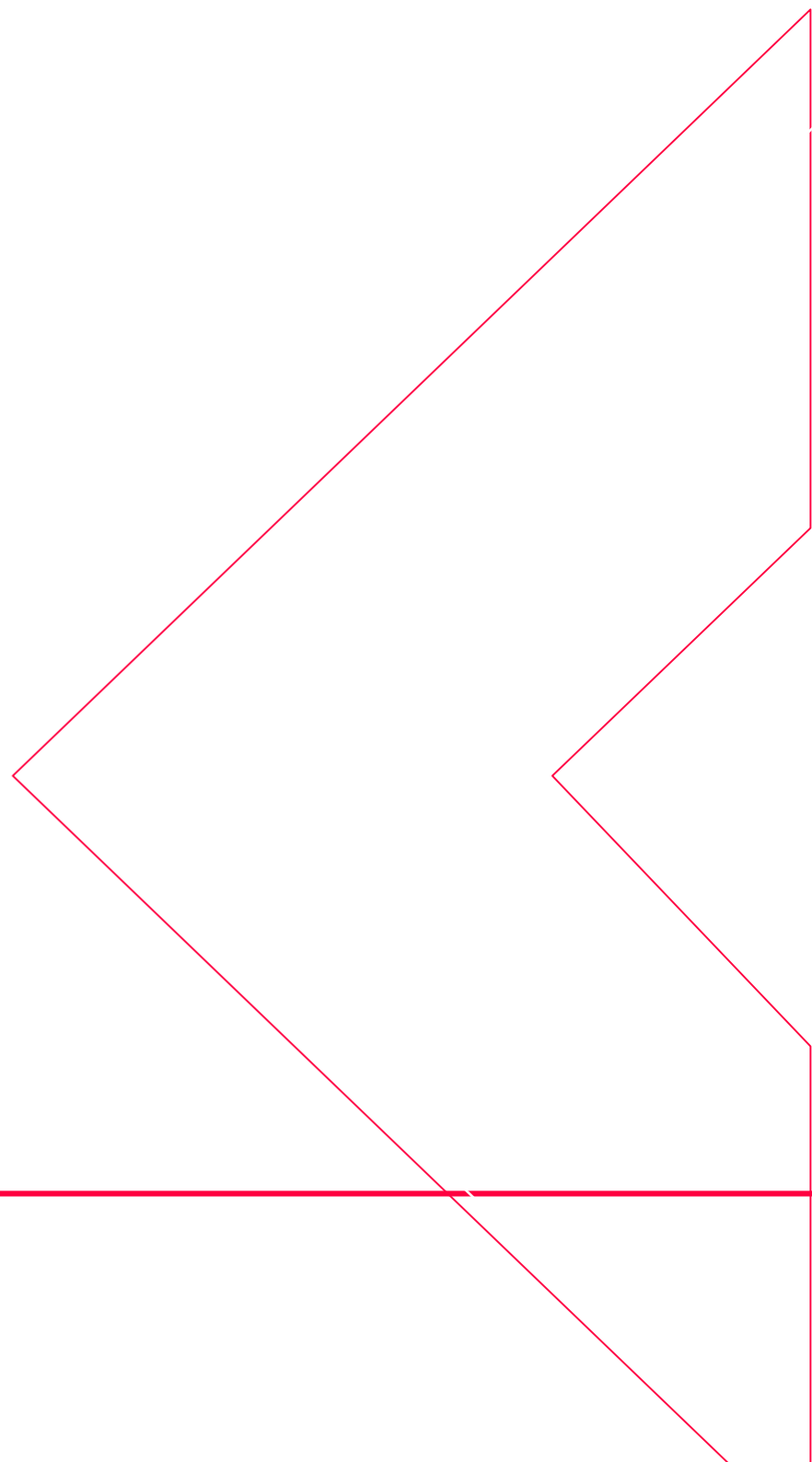


Рисунок 36 - Вид кнопки 'Main menu'

ПРАКТИЧЕСКИЕ ЗАДАЧИ



Задача 1



Создать программу в среде PyCharm, в которой пользователь два массива данных со следующими параметрами: 'a' = [1, 2, 3, 4, 5] и 'b' = [6, 7, 8, 9].

Необходимо найти максимальные элементы в этих массивах и вывести на консоль как содержимое массивов, так и максимальные элементы.

Решение



Напишем код для решения данной практической задачи и посмотрим на вывод:

```
pythonProject3 - main.py
pythonProject3 > main.py
1 # ВВОД ЗНАЧЕНИЙ
2 a = str(input('Введите a: '))
3 b = str(input('Введите b: '))
4 # ПОИСК МАКСИМАЛЬНЫХ ЗНАЧЕНИЙ
5 a_max = max(a)
6 b_max = max(b)
7 # ВЫВОД ДАННЫХ
8 print('Массив a:', a)
9 print('Массив b:', b)
10 print('Максимальный элемент в a:', a_max)
11 print('Максимальный элемент в b:', b_max)
```

Run: main ×

```
Введите a: 12345
Введите b: 6789
Массив a: 12345
Массив b: 6789
Максимальный элемент в a: 5
Максимальный элемент в b: 9
```

Version Control Python Packages TODO Python Console Problems Terminal Services
Download pre-built shared indexes: Reduce the indexing time and CPU load with pre-built Python packages shared indexes // Always download //

Задача 2.



Создать программу в среде PyCharm, в которой пользователь два массива данных со следующими параметрами: 'x' = [2, 4, 6, 8] и 'y' = [1, 3, 5, 7]. Необходимо найти минимальные элементы в этих массивах и вывести на консоль как содержимое массивов, так и минимальные элементы.

Решение



Для решения данной задачи понадобится следующий код:

```
pythonProject3 - main.py
pythonProject3 - main.py
1 # ВВОД значений
2 x = str(input('Введите x: '))
3 y = str(input('Введите y: '))
4 # поиск минимальных значений
5 x_min = min(x)
6 y_min = min(y)
7 # вывод данных
8 print('Массив x:', x)
9 print('Массив y:', y)
10 print('Минимальный элемент в x:', x_min)
11 print('Минимальный элемент в y:', y_min)
```

Run: main x

```
Введите x: 2468
Введите y: 1357
Массив x: 2468
Массив y: 1357
Минимальный элемент в x: 2
Минимальный элемент в y: 1
```

Version Control Python Packages TODO Python Console Problems Terminal Services

End of statement expected

Вопросы



1. Для чего используется Figma, при работе с приложениями?
2. Что такое Dev Mode?
3. Какую основную функцию выполняет операция $\max(x)$?
4. Что такое редактор и какими они бывают?
5. Какой этап разработки является самым главным в среде PyCharm?