

Обзор IDE PyCharm

Что такое PyCharm?

PyCharm — интегрированная среда разработки, с полным набором средств предназначенных для разработки на языке Python.

Функции и возможности PyCharm

PyCharm предоставляет большой набор инструментов из “коробки”, как и многие другие продукты от компании разработчиков JetBrains.

Сразу после установки он имеет в себе:

- встроенный визуальный отладчик;
- возможность запуска в любом окружении;
- интерактивную консоль и встроенный терминал;
- инструменты для работы с базами данных;
- инструменты для работы веб-разработки;

и это только малая часть из доступных инструментов.

PyCharm поддерживает все основные реализации языка Python: Python 2.x и 3.x, Jython, IronPython, PyPy и Cython, что позволяет ему делать следующее:

- подсвечивать синтаксис;
- проверять код на ошибки с их указанием ещё до компиляции и запуска;
- проводить автодополнение кода;
- обеспечивать навигацию по коду и возможность просматривать структуру кода;
- проводить рефакторинг кода;

и многое другое.

Также важно отметить, что PyCharm это кроссплатформенная IDE, возможности которой пользователь всегда может расширить в зависимости от своих требований, благодаря большому количеству различных плагинов.

Необходимое программное и аппаратное обеспечение

PyCharm может быть установлен 2-мя способами, либо скачиванием установочного файла с официального раздела сайта GetBrains, либо через их приложение ToolBox. Не зависимо от выбора способа, для начала работы с PyCharm и написания первой программы, не потребуется установки какого-либо дополнительного ПО, т.к. всё необходимое устанавливается вместе с установкой самого приложения.

Системные требования:

1. ОС:

- 64-битная версия Microsoft Windows 10, 8;
- macOS 10.14 или выше;
- Любой дистрибутив Linux, поддерживающий Gnome, KDE или Unity DE.

2. Любой современный многоядерный CPU

3. RAM: не менее 4 ГБ, рекомендуется 8 ГБ

4. Не менее 10 ГБ свободного места на диске.

Форматирование кода на PyCharm

Данная IDE, как и многие другие имеет обширный функционал, который автоматически помогает пользователю с форматированием кода, при его непосредственном написании. Из таких функций можно выделить автоматическую табуляцию и расстановку пробелом в различных элементах кода, которая выставляется, в зависимости от используемых структур, подсветку синтаксиса и возможность просмотра начала и конца используемых структур, автозамену текста, в случае ошибочного написания кода на русском языке и многие другие функции, который делают код более наглядным и удобным для чтения.



```
1 from urllib import request
2 from html.parser import HTMLParser
3
4 url = 'https://en.wikipedia.org/wiki/Normal_distribution'
5
6 page = request.urlopen(url).read().decode('utf-8')
7
8
9 class MyHTMLParser(HTMLParser):
10     inside_a = False
11
12     def handle_starttag(self, tag, attrs):
13         if tag == 'a':
14             self.inside_a = True
15             attrs = dict(attrs)
16             if 'href' in attrs and (attrs["href"][0] != "#") and (attrs["href"][0] != "/"): print(attrs['href'])
17             if 'href' in attrs and (attrs["href"][0] == "/"):
18                 print("https://en.wikipedia.org" + attrs["href"])
19
20
21 parser = MyHTMLParser()
22 parser.feed(page)
```

Отладка кода на PyCharm

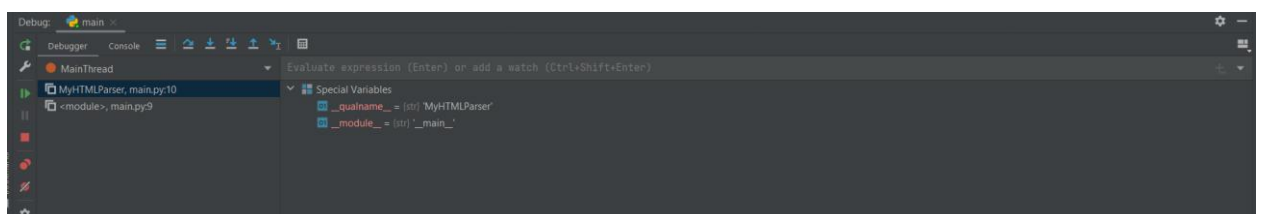
Отладка программы — это пошаговое выполнение вашей программы с целью обнаружения ошибок и проверки правильности кода.

Наша IDE имеет удобный отладчик, который позволяет отслеживать любые изменения в переменных, начиная с момента, который выберет пользователь. Чтобы начать отладку необходимо указать строку кода (так называемую точку останова), с которой отладчику необходимо начать работу, отметив её простым нажатием рядом с номером (она выделится красным кружком и будет выделена).



```
8
9 class MyHTMLParser(HTMLParser):
10     inside_a = False
11
12     def handle_starttag(self, tag, attrs):
13         if tag == 'a':
```

Затем либо через кнопку запуска отладки либо через комбинацию клавиш shift+F9 запускается сам процесс отладки, после чего появляется специальное окно, через которое можно наблюдать за изменением отдельных переменных, добавив их в список, введя их названием в соответствующую строку.



```
Debugger
Main Thread
MyHTMLParser.main.py:10
<module>.main.py:9
Special Variables
__qualname__ = (str) 'MyHTMLParser'
__module__ = (str) '__main__'
```

Также нельзя не отметить, что сам обход кода может проводиться 2-мя способами:

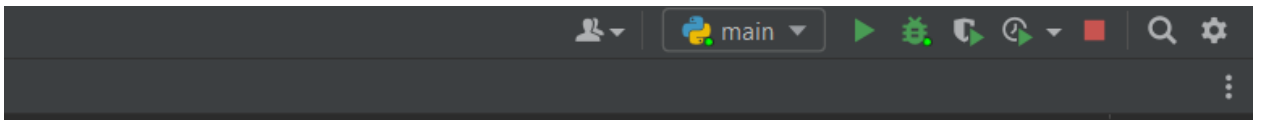
- Шаг с заходом (отладчик будет «заходить» внутрь всех функций и других конструкций, если их встретит);
- Шаг с обходом (отладчик не будет «заходить» внутрь всех функций и других конструкций, а просто выполнит весь код функции без остановки и продолжит выполнение дальше).

Некоторой особенностью и большим плюсом является возможность работы отладчика с несколькими точками остановки, что позволяет работать с конкретным куском кода.

```
11
12 def handle_starttag(self, tag, attrs):
13     if tag == 'a':
14         self.inside_a = True
15         attrs = dict(attrs)
16         if 'href' in attrs and (attrs["href"][0] != "#") and (attrs["href"][0] != "/"): print(attrs['href'])
17         if 'href' in attrs and (attrs["href"][0] == "/"):
18             print("https://en.wikipedia.org" + attrs["href"])
19
20
```

Запуск и компиляция кода

Как и во многих других IDE компиляция кода происходит сразу после его непосредственного запуска, без необходимости в каком либо стороннем аппаратном или пользовательском вмешательстве, что обеспечивает простоту использования данной IDE. Однако, как было сказано ранее, любой код может быть запущен не только целиком, но и отдельными кусками, с помощью выбора точек остановки, с которых отладчик будет начинать свою работу, для их непосредственного дебага.



Но PyCharm имеет очень широкий спектр возможностей, связанных с веб-разработкой, который обеспечивает поддержку популярных веб-фреймворков. Поддерживает функцию Live-edit, которая компилирует и сохраняет код при его непосредственном изменении и позволяет сразу же отслеживать полученные изменения на веб-странице.