**Инициализация объектов, метод Initialize**

[Java Syntax](https://javarush.ru/quests/QUEST_JAVA_SYNTAX)

[Уровень 5](https://javarush.ru/quests/lectures?quest=QUEST_JAVA_SYNTAX&level=5), Лекция 6

— Хочу рассказать тебе об инициализации объектов. Когда объект создаётся – его переменным нужно присвоить стартовые данные. Чтобы не было ситуаций, когда ты обращаешься к объекту, а внутри у него нет нужной ему информации для правильной работы.

— Рассмотрим для примера объект типа File (файл). Минимальной необходимой информацией для файла является его имя. Создать файл без имени – это бессмыслица.

— Допустим, мы пишем свой класс для работы с файлами — MyFile, какая информация нужна каждому объекту этого класса?

— Имя файла, с которым этот объект будет работать?

— Правильно. Для этого добавим в наш класс метод **initialize**(). Это будет выглядеть примерно так:

Пример:

class MyFile

{

private String filename = null;

public void **initialize**(String name)

{

**this.**filename = name;

}

…

}

— Мы добавили метод **initialize**, чтобы с объектом можно было работать – вызывать его методы. Это можно делать сразу после вызова метода **initialize**. Если с объектом работать нельзя, его называют ***невалидным*** (invalid), если можно – ***валидным*** (valid). Основная задача метода initialize – передать в объект все необходимые данные, чтобы сделать его валидным.

— Ясно.

— Теперь усложним задачу. Вернее упростим. Как посмотреть. Представим, что другому программисту, который будет использовать наш класс, удобнее передавать в него не полное имя файла, а директорию и короткое имя файла. Мы бы смогли реализовать эту функциональность с помощью ещё одного метода **initialize** (Java позволяет создавать несколько методов с одинаковыми именами). Тогда наш класс стал бы выглядеть так:

Пример с двумя методами initialize

class MyFile

{

private String filename = null;

public void **initialize**(String name)

{

**this.**filename = name;

}

public void initialize(String folder, String name)

{

**this.**filename = folder + name;

}

…

}

— А ещё, часто нужно создать временную копию файла рядом с текущим.

— Мы можем сделать метод для этого случая?

— Конечно, вот смотри:

Создание файла рядом с текущим файлом:

class MyFile

{

private String filename = null;

public void **initialize**(String name)

{

this.filename = name;

}

public void **initialize**(String folder, String name)

{

this.filename = folder + name;

}

// Файл filename будет находиться в той же директории, что и file.

public void **initialize**(MyFile file, String name)

{

this.filename = file.**getFolder**() + name;

}

…

}

— И что, я могу сделать таких методов сколько угодно?

— Разумные пределы, конечно, есть. Но, в принципе, да, сколько угодно.

— А когда надо вызывать метод initialize?

— Сразу после создания объекта, чтобы перевести его в валидное состояние:

|  |
| --- |
| **Примеры** |
| MyFile file = new MyFile();  file.initialize("c:\data\a.txt");  String text = file.readText(); |
| MyFile file = new MyFile();  file.initialize("c:\data∖", "a.txt");  String text = file.readText(); |
| MyFile file = new MyFile();  file.initialize("c:\data\a.txt");  MyFile file2 = new MyFile();  file2.initialize(file, "a.txt");  String text = file2.readText(); |

2

Задача

Java Syntax,  5 уровень,  6 лекция

Набираем код Ӏ Java Syntax: 5 уровень, 6 лекция

Java Syntax: 5 уровень, 6 лекция. Иногда думать не надо, строчить надо! Как ни парадоксально звучит, порой пальцы «запоминают» лучше, чем сознание. Вот почему во время обучения в секретном центре JavaRush вы иногда встречаете задания на набор кода. Набирая код, вы привыкаете к синтаксису и зарабатываете немного материи. А ещё — боретесь с ленью.

— А что ещё за метод getFolder()?

— По логике этот метод должен возвращать строку с именем папки, в которой находится наш файл. Но мы его тут не писали, он чисто для фона.