**Создание объекта. Порядок вызова конструкторов**

[Java Core](https://javarush.ru/quests/QUEST_JAVA_CORE)

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

— Привет, Амиго! Сейчас я расскажу тебе о процессе создания объекта.

— А что там сложного, дядя Риша? Написал new имя класса, указал правильный конструктор и готово!

— Это так. Но что при этом происходит внутри объекта?

— А что там происходит?

— А вот что! Объект создается в несколько этапов.

***1)****Сначала выделяется память под все переменные – поля класса.*

***2)****Затем идет инициализация базового класса.*

***3)****Потом всем переменным присваиваются значения, если они указаны.*

***4)****И наконец, вызывается конструктор.*

— Выглядит не очень сложно: сначала переменные, затем конструктор.

— Давай посмотрим, как это будет работать на примере двух классов:

|  |  |
| --- | --- |
| **Код** | **Описание** |
| class Pet  {  int x = 5, y = 5; ←-  int weight = 10; ←-  Pet(int x, int y)  {  this.x = x; ←-  this.y = y; ←-  }  }  class Cat extends Pet  {  int tailLength = 8; ←-  int age;  Cat(int x, int y, int age)  {  super(x, y); ←-  this.age = age; ←-  }  } | Мы создали два класса: Pet(животное) и Cat(кот).  В классе Cat мы видим явный вызов конструктора базового класса. **Он всегда должен быть в первой строке конструктора.**  Вот что произойдет после выделения памяти: 18 – вызов конструктора базового класса 3, 4 – инициализация переменных в Pet 8, 9 – отработает код конструктора Pet  далее начнется процесс инициализации класса Cat 14 – инициализация переменных в Cat 19 – отработает код конструктора Cat |
| public static void main(String[] args)  {  Cat cat = new Cat (50, 50, 5);  } |  |

— Что-то немного запутанно. Почему так сложно?

— На самом деле не сложно, если знать что на самом деле происходит:  
**Если у класса нет ни одного конструктора, он будет создан автоматически.**

|  |  |
| --- | --- |
| **Конструктор по умолчанию** |  |
| class Cat  {  int x = 5;  int y = 5;  } | class Cat  {  int x = 5;  int y = 5;  public Cat()  {  }  } |

**Если не вызываешь конструктор базового класса, его вызов будет добавлен автоматически.**

|  |  |
| --- | --- |
| **Вызов конструктора базового класса** |  |
| class Pet  {  public String name;  } | class Pet extends Object  {  public String name;  public Pet()  {  super();  }  } |
| class Cat extends Pet  {  int x = 5;  int y = 5;  } | class Cat extends Pet  {  int x = 5;  int y = 5;  public Cat()  {  super();  }  } |

Инициализация переменных класса происходит в конструкторе.

|  |  |  |
| --- | --- | --- |
| **Инициализация переменных класса** |  | |
| class Cat  {  int x = 5;  int y = 5;  } | class Cat  {  int x;  int y;  public Cat()  {  super();  this.x = 5;  this.y = 5;  }  } | |
| **Как все это происходит на самом деле** | |  | |
| class Pet  {  int x = 5, y = 5;  int weight = 10;  Pet(int x, int y)  {  this.x = x;  this.y = y;  }  }  class Cat extends Pet  {  int tailLength = 8;  int age;  Cat(int x, int y, int age)  {  super(x, y);  this.age = age;  }  } | | class Pet extends Object  {  int x;  int y;  int weight;  Pet(int x, int y)  {  //вызов конструктора базового класса  super();  //инициализация переменных  this.x = 5;  this.y = 5;  this.weight = 10;  //вызов кода конструктора  this.x = x;  this.y = y;  }  }  class Cat extends Pet  {  int tailLength;  int age;  Cat(int x, int y, int age)  {  //вызов конструктора базового класса  super(x, y);  //инициализация переменных  this.tailLength = 8;  //вызов кода конструктора  this.age = age;  }  } | |

4

Задача

Java Core,  5 уровень,  5 лекция

Набираем код Ӏ Java Core: 5 уровень, 5 лекция

Иногда думать не надо, строчить надо! Как ни парадоксально звучит, порой пальцы «запоминают» лучше, чем сознание. И даже на втором квесте иногда полезно просто понабирать код. Вперёд!

— Теперь намного понятнее: сначала базовый класс, затем переменные вне конструктора, затем вызов кода конструктора.

— Молодец, Амиго, именно так!