

IDEA: debug,точки останова

Java Collections
9 уровень, 3 лекция

ОТКРЫТА

— Привет, Амиго!

— Привет, Элли! У тебя новая прическа? Тебе очень идет!

— Да? Спасибо!



Сегодня я расскажу тебе про работу в IntelliJ IDEA.

— Так я уже работаю в ней и довольно давно.

— Да, я знаю, поэтому и хочу тебе рассказать про некоторые вещи, которые значительно упрощают жизнь.

Первое и самое главное, что должен уметь каждый разработчик – это отладка программы. Или как еще называют – дебаг. Баг по-английски – жук – сленговое название ошибок в программе.

В IntelliJ IDEA можно запустить приложение в двух режимах.

Кнопки	Режимы
	Обычный запуск программы
	Запуск программы в режиме отладки (дебага)

— Ага. Кнопка в виде жука – это дебаг. Прикольно придумано.

— **Самое важное!** При запуске в режиме отладки ты можешь выполнять приложение пошагово. По одной строчке за раз.

А самая важная часть дебага – это **точки останова**.

На любой строчке кода ты можешь поставить точку останова – **BreakPoint**. Программа, запущенная в режиме отладки, дойдёт до этой точки и остановится. Чтобы поставить BreakPoint, надо поставить курсор на нужную строку и нажать Ctrl+F8. Чтобы убрать – снова Ctrl+F8.

Чтобы продолжить исполнение программы до следующей точки остановки, надо нажать F5.

Чтобы продолжить исполнение программы пошагово (построчно), надо нажать F7 или F8.

При этом, при нажатии F7 программа будет заходить в методы, когда они вызываются.

При нажатии F8 вызов метода будет считаться за 1 шаг – программа не будет заходить в методы.

— А можно рассказать об этом нюансе более подробно?

— Конечно. Вот скопируй себе код, а я на его примере объясню, что надо делать:

Код	
1	package com.javarush.test;
2	
3	public class MainClass
4	{

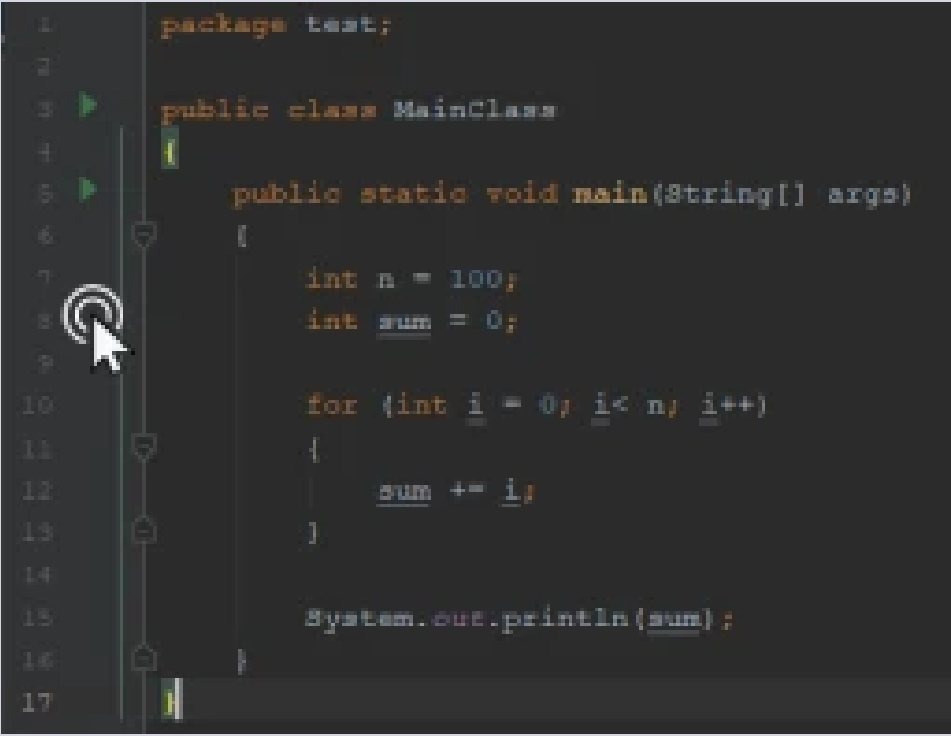
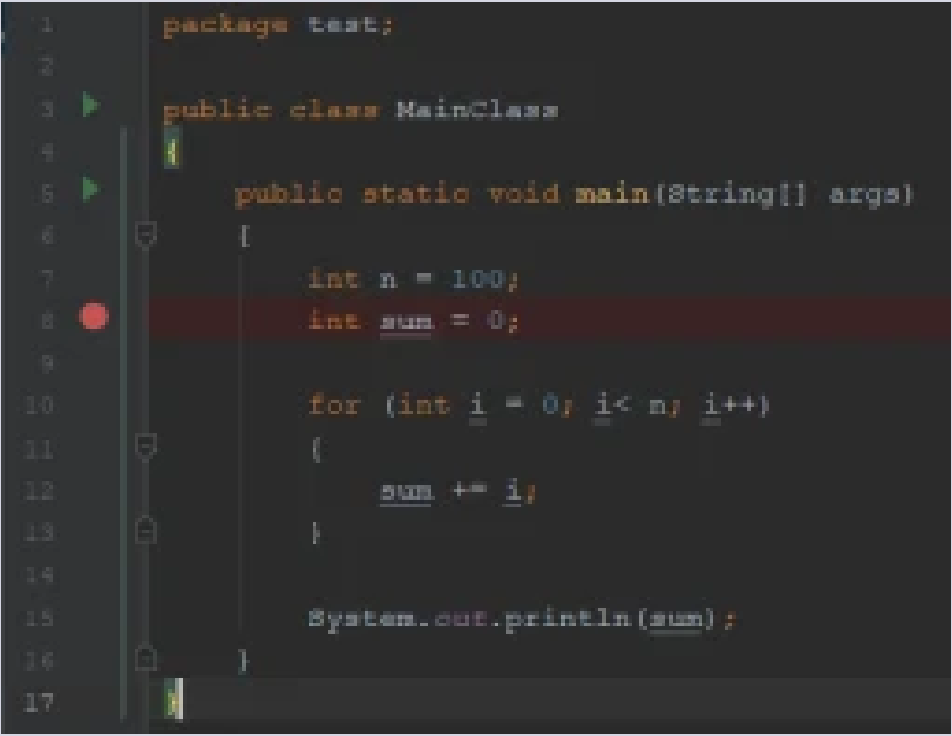
```
5      public static void main(String[] args)
6      {
7          int n = 100;
8          int sum = 0;
9
10         for (int i = 0; i< n; i++)
11         {
12             sum += i;
13         }
14
15         System.out.println(sum);
16     }
17 }
```

В этом примере мы просто считаем сумму чисел от 0 до n.

Как поставить точку остановки – BreakPoint

Вариант 1 – стать курсором на нужную строку и нажать Ctrl+F8.

Вариант 2 – кликнуть слева от нужной строки.

Как поставить точку остановки	Результат
	

Чтобы убрать BreakPoint – кликни на красный кружочек или нажми Ctrl+F8.

Теперь запускаем программу кликом мышкой на кнопке дебаг.

Должно получится что-то типа такого:

```
1 package test;
2
3 public class MainClass
4 {
5     public static void main(String[] args)
6     {
7         int n = 100;    n: 100
8         int sum = 0;
9
10        for (int i = 0; i < n; i++)
11        {
12            sum += i;
13        }
14
15        System.out.println(sum);
16    }
17 }
```

Весь код программы выполненлся до синей строки. Строка, выделенная синим цветом, еще не выполнилась.

Нажми F8 и выполни ее. Должно получится как на картинке ниже:

```
1 package test;
2
3 public class MainClass
4 {
5     public static void main(String[] args)
6     {
7         int n = 100;    n: 100
8         int sum = 0;    sum: 0
9
10        for (int i = 0; i < n; i++)    n: 100
11        {
12            sum += i;
13        }
14
15        System.out.println(sum);
16    }
17 }
```

Красная – точка остановки – BreakPoint

Синяя – текущая строка отладки/дебага

Давай заменим число 100 в коде на 5 и попробуем выполнить всю программу пошагово. Вот какая будет последовательность шагов:

<pre>1 package test; 2 3 public class MainClass 4 { 5 public static void main(String[] args) 6 { 7 int n = 5; n: 5 8 int sum = 0; 9 10 for (int i = 0; i < n; i++) 11 { 12 sum += i; 13 } 14 15 System.out.println(sum); 16 } 17 }</pre>	<pre>1 2 4 6 8 10 12 3 5 7 9 11 13 14</pre>
--	--

Первый шаг – это строчка выделенная красным.

Пустые строки пропускаются, так же как и скобки – там нет никакого кода.

Теперь давай немного усложним программу, и я покажу тебе разницу между F7 и F8.

1package test;

2

3▶public class MainClass

4{

5▶public static void main(String[] args)

6{

7int sum5 = sum(n: 5);

8System.out.println(sum5);

9

10int sum7 = sum(n: 7);

11System.out.println(sum7);

12

13}

14

15@public static int sum(int n)

16{

17int sum = 0;

18

19for (int i = 0; i < n; i++)

20{

21sum += i;

22}

23

24return sum;

25}

26|

27}

F7

12

1516

35

317

46812182032

57913192133

1434

F8

12

34

5

Если ты нажимаешь F8, то выполняешь текущую строку за 1 шаг.

Если нажимаешь F7 и в текущей строке вызов метода, ты «заходишь» туда и выполняешь его пошагово.

- Т.е. разница в том – заходим мы в метод или нет.

- Ага.

- А я могу комбинировать F7 и F8? Т.е. какие-то методы пропускаю, которые мне не интересны, а в какие-то захожу?

- Да.

−

+30

+

Комментарии (51)

популярные

новые

старые

JavaCoder

Введите текст комментария

Ответить

Владислав

Backend Developer

27 июля 2020, 16:50

...

Методы какие-то. Надеюсь, что в следующей лекции расскажут про это что-нибудь)

Ответить

+13

Pig Man

Главная свинья в Свинарнике

24 февраля 2021, 18:27

...

Что такое IntelliJ IDEA?

Ответить

+2

Показать еще комментарии

ОБУЧЕНИЕ

- Курсы программирования
- Курс Java
- Помощь по задачам
- Подписки
- Задачи-игры

СООБЩЕСТВО

- Пользователи
- Статьи
- Форум
- Чат
- Истории успеха
- Активности

КОМПАНИЯ

- О нас
- Контакты
- Отзывы
- FAQ
- Поддержка



JavaRush — это интерактивный онлайн-курс по изучению Java-программирования с нуля. Он содержит 1200 практических задач с проверкой решения в один клик, необходимый минимум теории по основам Java и мотивирующие фишки, которые помогут пройти курс до конца: игры, опросы, интересные проекты и статьи об эффективном обучении и карьере Java-девелопера.

ПОДПИСЫВАЙТЕСЬ

ЯЗЫК ИНТЕРФЕЙСА

Русский

▼

