String manipulation with reflection

exercises and hints

Bogdan Stroe, Daniel Ciocîrlan

Oracle

May 18, 2016

TODO 0: import the file

- Eclipse or your favorite IDE
- (recommended) new Java project
- drag-and-drop StringManipulator.java
- or create a new StringManipulator class and paste the code

Manipulating Strings: setting a character

- Strings are *immutable*
    - what does it mean?

Manipulating Strings: setting a character

- Strings are *immutable*
    - what does it mean?
- we can *only* modify internal values through reflection

Manipulating Strings: setting a character

- Strings are *immutable*
  - what does it mean?
- we can *only* modify internal values through reflection

Steps:

- get the `Class<String>` object for the String type
- get a `Field` object for the internal field you want to modify
  - a Field is just an accessor object
  - its purpose is to extract and inject values into "real" objects
  - in our case, the name of the internal field is "value"
- get the actual content of the "value" field of the String
  - use the Field object
- set the character in the char array

Manipulating Strings: appending a character

- What happens when we usually append a char to a String?

Manipulating Strings: appending a character

- What happens when we usually append a char to a String?
- *only* reflection can avoid the creation of new Strings

Manipulating Strings: appending a character

- What happens when we usually append a char to a String?
- *only* reflection can avoid the creation of new Strings

Steps (most are the same as before):

- get the `Class<String>` object for the String type
- get a `Field` object for the internal field you want to modify
  - a Field is just an accessor object
  - its purpose is to extract and inject values into "real" objects
  - in our case, the name of the internal field is "value"
- create a new (longer) char array, copy the contents and the new char
- *set* the content of the "value" field of your String
  - use the Field object