Classes - Testing your classes

Make sure your classes do what you think
Classes are there for creating objects

- Most classes in Java are blueprints for creating objects
- Objects have state and behaviour
- How do you know that they have a legal state?
- How do you know that their state makes sense?
- How do you know that they behave the way we want?
Small example

- We are writing a membership management system
- We have a class representing members in some club
- Members have a name and an email
- The email can be changed
- Members should only store legal email addresses
package net.supermegacorp.orgmanager;

public class Member {

    private String name;
    private String email; // null means no email
    private static final String separator = ";";

    public Member(String name, String email) {
        this(name);
        setEmail(email);
    }

    public Member(String name) {
        this.name = name;
    }
}
public void setEmail(String email) {
    if (email.contains("@")) {
        this.email = email;
    }
}

public String toString() {
    return name + separator + email;
}

public String name() {
    return name;
}

public String email() {
    return email;
}

public static String getSeparator() {
    return separator;
}
}
Thinking about what can go wrong

- Do we allow only legal emails?
  - How can we test this?
- What happens if we try to change the email to `null`?

```java
public void setEmail(String email) {
    if (email.contains("@")) {
        this.email = email;
    }
}
```
Legal email?

- Let’s pretend for the sake of simplicity that we are satisfied with the check for the @ sign (this is of course a poor email validity check!)
- To test this simple algorithm, we write a small program which creates a few Member objects, and try to change their email to a “legal” one and also an “illegal” (without @ sign)
package net.supermegacorp.orgmanager.test;

import net.supermegacorp.orgmanager.Member;

public class MemberTest {

    public static void main(String[] args) {
        Member ada = new Member("Ada", "ada@lovelace.edu");
        System.out.println(ada);
        ada.setEmail("ada_AT_lovelace.edu");
        System.out.println(ada);
    }
}

What will it print?
package net.supermegacorp.orgmanager.test;

import net.supermegacorp.orgmanager.Member;

public class MemberTest {

    public static void main(String[] args) {
        Member ada = new Member("Ada", "ada@lovelace.edu");
        System.out.println(ada);
        ada.setEmail(null);
        System.out.println(ada);
    }
}

What will it print?
Think about normal values and extreme values

- Changing email to null didn’t work so well - null is an excellent test case :)
- Think about other extreme cases
- Numbers?
  - Negative numbers
  - Ridiculously large numbers
  - Zero is sometimes impossible or wrong
Test program with printouts

- You can write your tests so that they print a report etc
- Will create a lot of IF statements and reports can be long and hard to read

Suggestion:

- Short summary at the end (only print failures on the way)
- Crash the program at first failed test, so that you must fix the problem before running the tests again
- Use assertions instead of IF statements
Short introduction to assertions

This is kind of noisy:

```java
if ( !ada.email().equals("ada@lovelace.edu")) {
    System.err.println("Ada's email " + ada.email() + " is faulty: ");
}
```

What we want to do is to make sure that ada’s email is correct, and crash the program with an error message instead. Enter assertions!

```java
assert email.equals("ada@lovelace.net") : "Ada's email was wrong";
```
Syntax of the assert statement

<boolean expression> : <error message as string>

Example from previous page:

```java
assert email.equals("ada@lovelace.net") : "Ada's email was wrong";
```

Note that if email is null, the above will crash, but since we write the test code, we should have control over that ;-)
You have to enable assertions when running Java will completely ignore assertions unless you tell it to enable it.

In order to enable assertions, add the flag -ea on the command line:

```
java -ea net.supermegacorp.orgmanager.test.MemberTest
```
Further reading

- http://www.vogella.com/tutorials/JUnit/article.html