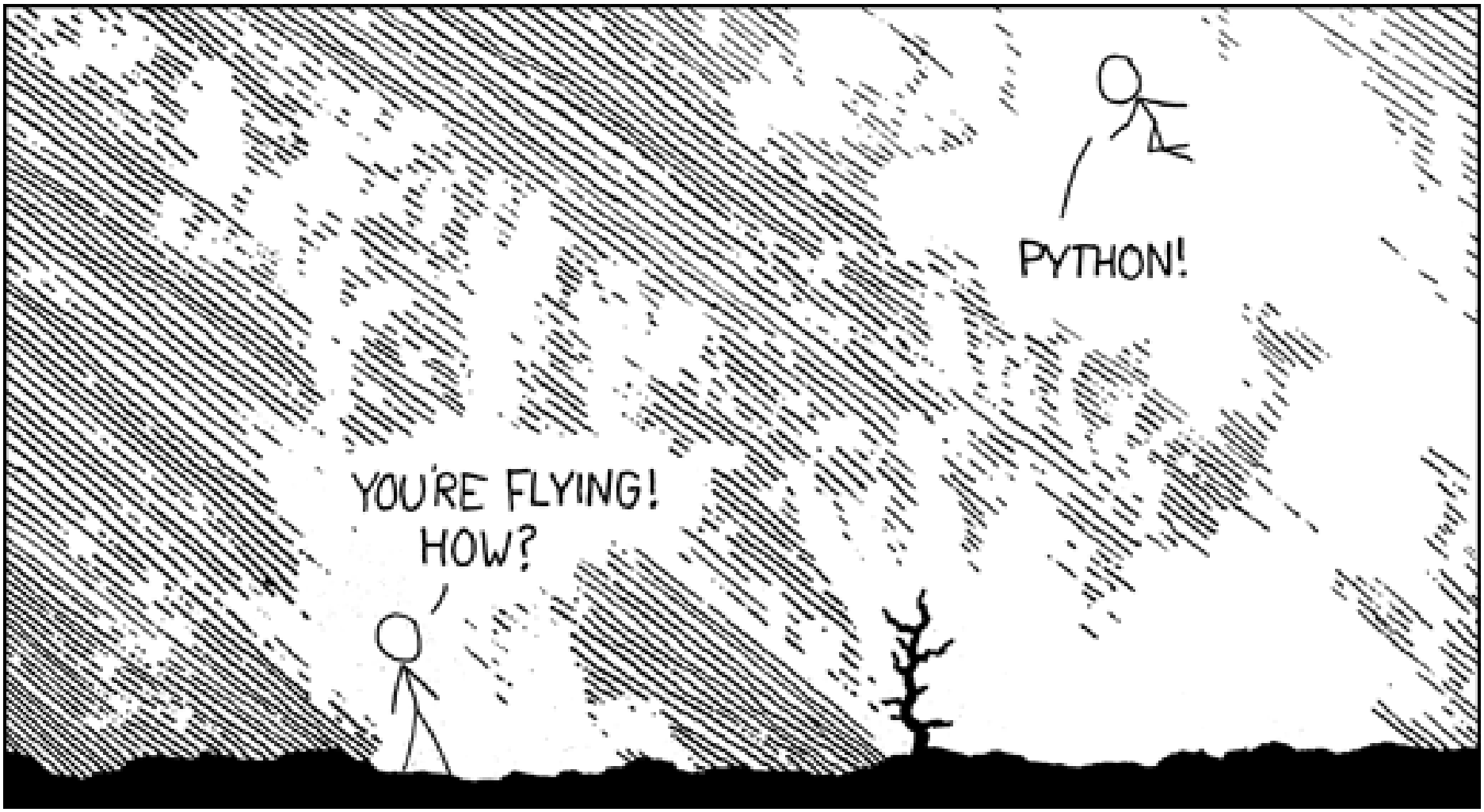
CompSci 230
Software Construction

Lecture Slides #2: Hello World! S1 2015

Agenda

- Topics:
  - "Hello world!" in Java and Python
  - Backward and forward compatibility
  - Syntax and semantics

xkcd 353: Python



xkcd 353: Python (2 of 2)

Hello World!

- `Hello.py` (Python source code):

```
print "Hello World!"
```

- Python has a shell -- a command-line interface which will execute a single line of code immediately after you type it. Very convenient!!

```
Python 3.4.3 (v3.4.3:9b73f1c3e601, Feb 24 2015, 22:43:06)
[MSC v.1600 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more
information.
>>> print "Hello World!"
  File "<stdin>", line 1
    print "Hello World!"
                        ^
SyntaxError: Missing parentheses in call to 'print'
>>> import antigravity
```

- Sigh. We're running Python 3.4.3, but the code was written for Python 2.

"Hello World!" in Python 3

- `Hello.py` (Python 3 source code):

```
print("Hello World!")
```

```
Python 3.4.3 (v3.4.3:9b73f1c3e601, Feb 24 2015, 22:43:06)
[MSC v.1600 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more
information.
>>> print("Hello World!")
Hello World!
>>>
```

- Python 3 isn't backward compatible.
  - It won't run "old code" correctly.
- Python 2.5 (2006) wasn't forward compatible.
  - In Python 2.6 (2008) and 2.7 (2010), it is *possible* to write code which can be translated (using 2to3) into code that will run correctly on Python 3 (2008-).
- A slow transition:
  - Some commonly-used libraries in Python 2 still haven't been ported to Python 3.

"Hello World!" in Java

- `Hello.java` (Java source code):

```
public class Hello {
    public static void main(String[] args) {
        System.out.println("Hello World!");
    }
}
```

- Python programs have much less gobbledygook than Java programs.
  - "language that is … made unintelligible by excessive use of technical terms" (OED online).
- Syntax is a set of rules defining what a compiler or interpreter "should accept" as a program.
  - You saw a syntax error message on slide #5.
  - Syntax is the "form" of a program.
- Semantics is the "meaning" of a program.
  - The semantics of a programming language define what a computer "should do" when it executes a program in that language.

More on Backward Compatibility

- Until 2011, Java had excellent backward compatibility.
  - Java programs that were compiled into bytecode in 2002 (J2SE 1.4) would still run in 2008, if you maintained a Java 1.4 runtime system on your platform.
    - Note: after 2008, Java 1.4 was no longer supported – no more security patches.
  - Java programs that were compiled into bytecode in 2006 (Java SE 6) would still run in 2012, if you maintained a Java 1.6 runtime system.
- Backward-compatibility of compiled code is
  - very desirable in software applications, because you can upgrade a system without affecting the software.
  - very undesirable for malware, because it is still dangerous on the upgraded system!
- In 2011, Oracle advised that
  - "keeping old and unsupported versions of Java on your system presents a serious security risk."
- The authors of Java are very careful to preserve backward-compatibility at the source-code level.
  - With few exceptions, old source code is syntactically correct on the current edition of Java.
  - Semantics are carefully controlled; but there are some changes across versions, so recompiled code should be tested to assure correct behaviour.
  - The main problem: you must rewrite any source code that imports an obsolete library.
- The authors of Python are now very aware of the importance of backward compatibility.
  - In April 2014, the end-of-life for Python 2 (2000-) was extended from 2014 to 2020, so that users who hadn't yet completed the port to Python 3 (2008-) would have enough time to do so.

Syntax and semantics of Java

- Java's syntax is similar to C/C++.
  - There's a *lot* of detail to learn, but it does make some sense (eventually ;-).
  - Once you have learned Java's syntax, you'll have a good head-start on C!
  - I won't attempt to teach Java syntax in my lectures.
    - Learning Java syntax is like learning how to spell words correctly in English: there's an awful lot to memorise, and only a few concepts.
    - The only way to learn Java syntax is by writing, and reading, a lot of Java programs!
      - The Java compiler will issue an error message when you "get it wrong".
      - Practice… and learn from your mistakes!
    - Don't aim for perfection.
      - You'll have Eclipse in the lab.
      - On a test or exam, yr mrkr cn prbbly ndrstnd wht y wrt vn f y mk fw rrrs.
- Python's semantics is similar to Java.
  - If you have a good working understanding of "what a Python program is supposed to do", you have a good head-start on Java semantics.
  - However, Python is weakly-typed, and Java is strongly-typed.
    - Learning Java's type system is a significant achievement for any programmer.
    - I'll devote quite a bit of lecture time to this concept, and the assignments should help.
    - You won't understand Java's type system in an hour, or in a day… but once you "get it", you'll be a competent Java programmer. Give it a go! We'll start on the next slide…

Dissection of a Java Class

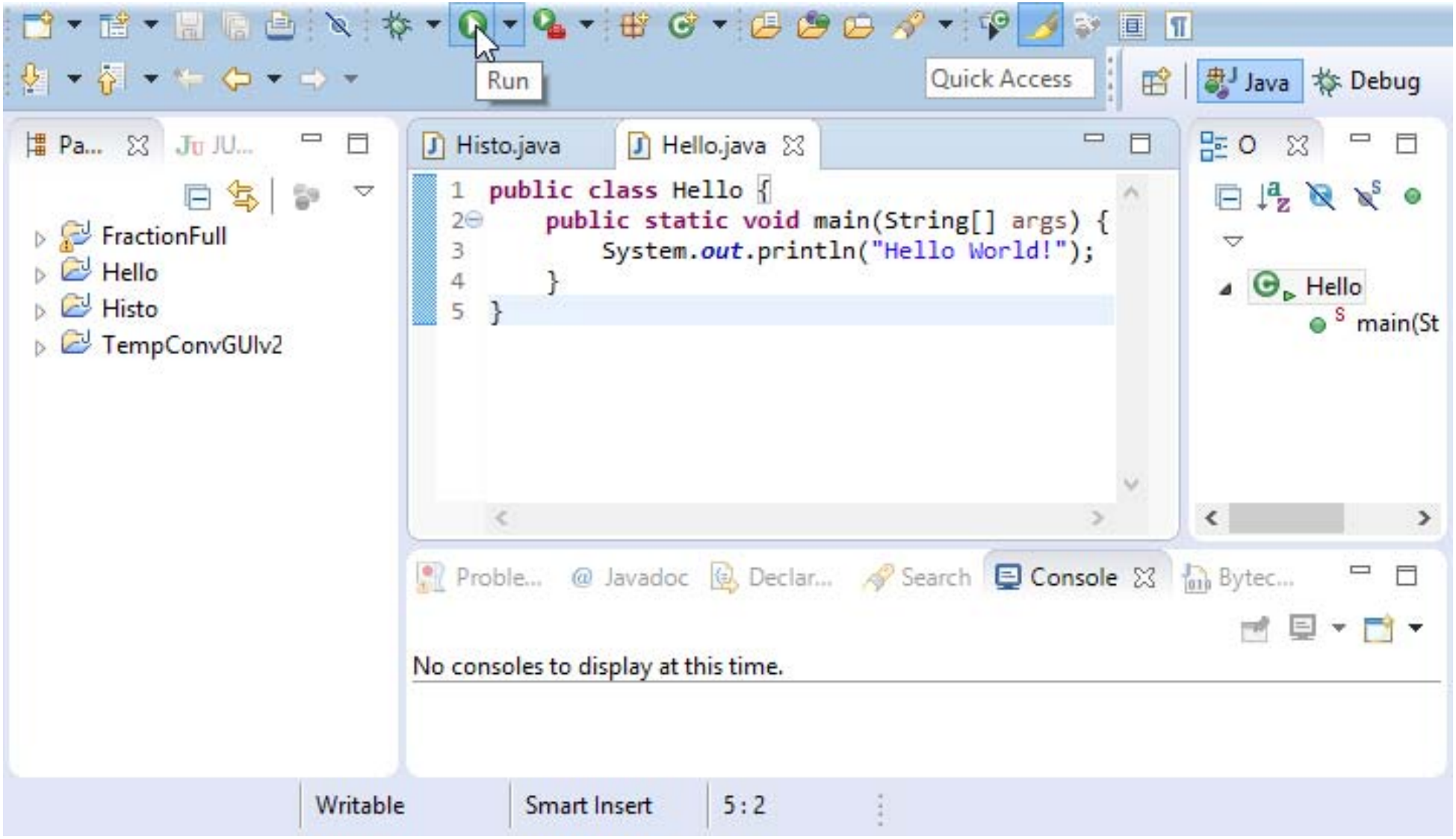
```
public class Hello {
        public static void main(String[] args) {
                System.out.println("Hello World!");
        }
}
```

This simple example illustrates a few very important rules:

1. Every Java program must define a class, all code is inside a class.
2. Everything in Java must have a type.
3. Every Java program must have a function called
   `public static void main(String[] args).`

[Section 2.4 of java4Python]

Try it in Eclipse!



Review

- "Hello world!" in Java and Python
  - Python 2 and Python 3 are different languages, with different syntax
  - Any version of Python has simpler syntax than any version of Java
  - Python and Java have similar semantics
  - Syntax and semantics: roughly, "form and meaning"
- Backward compatibility = designing new systems so they'll run old programs. Not always desirable:
  - Is a program malicious, or is it a "good" application?
  - Most Pythonistas agree that Python 3 is a big advance on Python 2, despite its lack of backward-compatibility.
- Forward compatibility = writing programs so that they'll run on future systems. Desirable but difficult!
  - (Predicting the future is outside the scope of this paper ;-)