2020 CI401
Introduction to programming

Week 1.01
Introduction

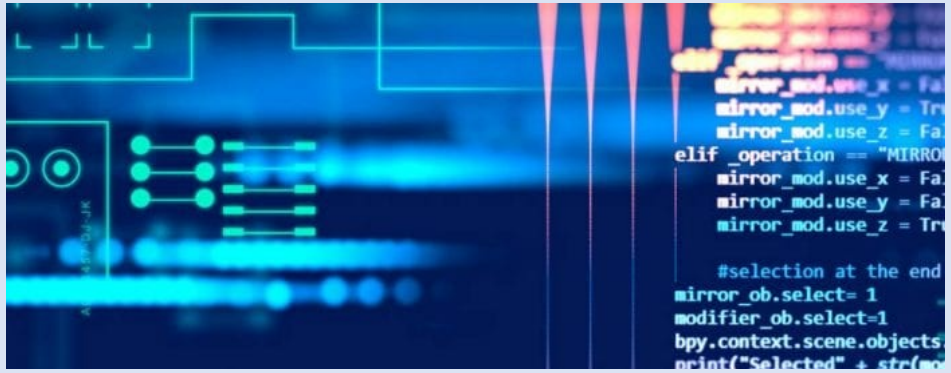
Dr Roger Evans

Module leader

6th October 2020

Module introduction

CI401:Introduction to Programming



- Is the backbone of your studies during your first year
- Teaches an essential skill for all flavours of computer science
- Includes all our main Computer Science course students (as a single cohort – this year, that's about 190 students)
- Develops *transferable* and *industry-oriented* coding skills

What do you learn?

- You will learn about programming in general, and how to code
- More than that, you learn how to code *well*
- The programming language we use is called *Java*. It is very widely used for teaching, research and in industry
- But we always say this is not a 'Java programming' course – it is a *programming course that uses Java*
- You learn skills that are *transferrable* – to other programming languages in other modules

How will you be taught? (2020 style)

- Live sessions
  - *Lectures* (1hr per week) – live session with lecturer, online
  - *Seminars* (1hr per week) – in course groups (about 40 students), guided activities and discussion, online
  - *Labs* (1hr per week) – individual lab work, with tutors present in labs on campus and available online for one-to-one support
- Independent learning
  - Lectures recorded for independent viewing
  - Additional pre-recorded learning topics for private study
  - Slides, notes and exercises

Module timetable - Semester 1

	Monday	Tuesday	Wednesday	Thursday	Friday
0900			Lab W202/online		
1000			Lab C107/online		
1100					Lab 107/online
1200	Seminar online		Lab C107/online		
1300					
1400		Lecture online			Lab C107/online
1500			Lab C107/online		
1600			Lab C107/online		

Your personal timetable

- You don’t usually see the whole timetable for a module like this
- You only see your *personal* timetable, with the particular sessions you have to attend
- You should make sure you have found your timetable and know what you are doing
- If you are having trouble, ask for help in a lab session

Module team contacts (semester 1)

Name	Role	Contact
Roger Evans	**Module leader** Lectures, seminars, labs	C507 (Cockcroft 5th floor) R.P.Evans@brighton.ac.uk (email) @Roger Evans (teams) @rogerevansbton (twitter)
Course administrators	**General questions**	CEMUGComputing@brighton.ac.uk
Ali Hamie	Seminars, labs	
Stelios Kapetanakis	Seminars, labs	
Khuong Nguyen	Seminars, labs	
Jarod Locke	labs	

Roger's 'office' policy

- Current university policy is to only be on campus for timetabled events and avoid unplanned/informal face to face meetings.
- I am available for online meetings any time I am free – send me an email or Teams chat message to arrange.
- If you need a (socially distanced) face to face meeting, we should arrange it in advance
- I am not available on Fridays.

Email: R.P.Evans@brighton.ac.uk

Teams: @Roger Evans

Tools

Key module resources

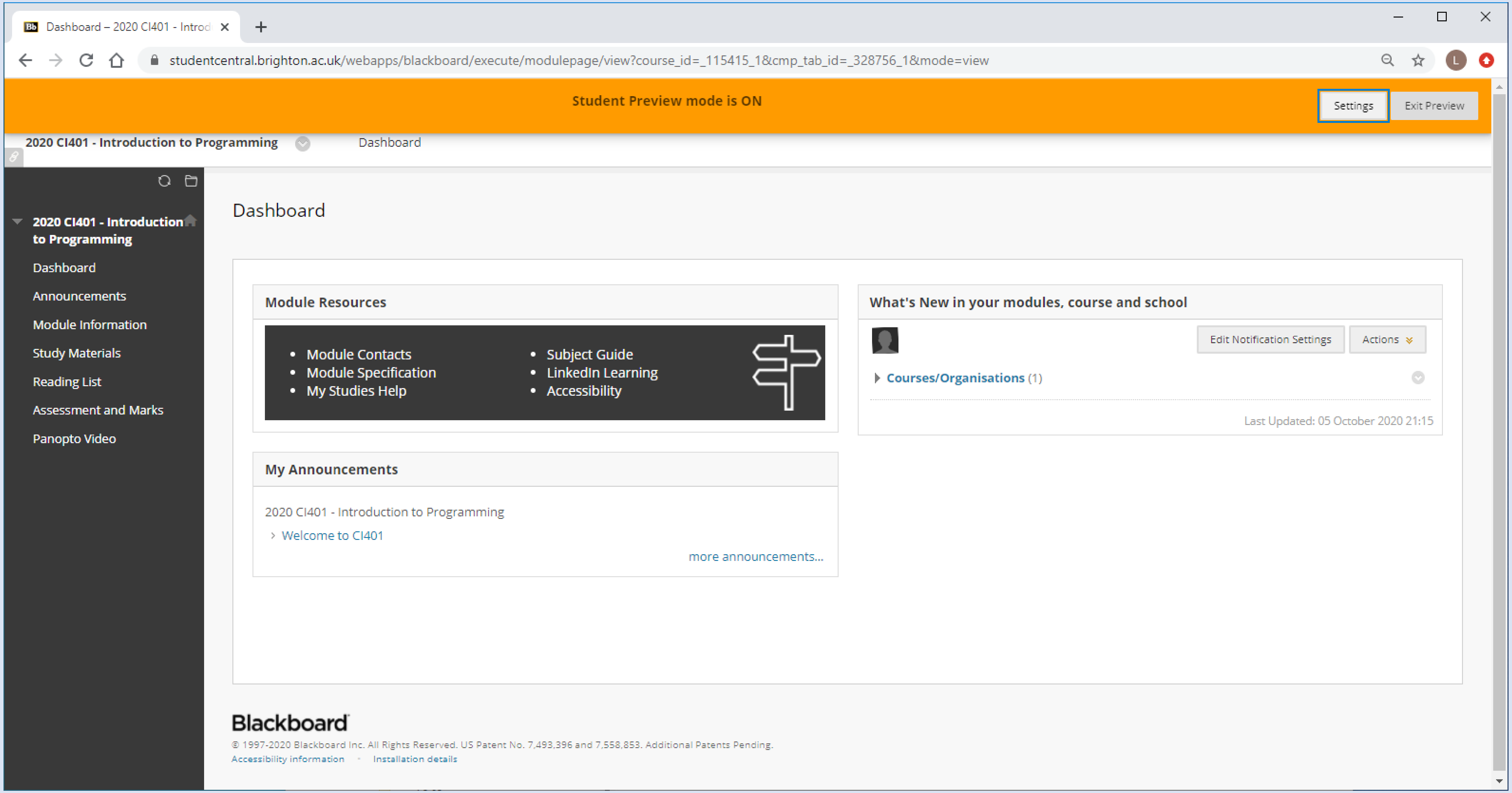
- mystudies.brighton.ac.uk
  - *2020 CI401 - Introduction to programming*
  - Main module area for announcements, resources, assessments, results etc.
- Microsoft Teams
  - *Group-2020 CI401 - Introduction to programming*
  - Teams area for live online sessions (lectures, seminars and labs) and chat streams
  - Access code: *fjfsjqo*

MyStudies

- MyStudies is the virtual learning environment (VLE) we use to support learning and teaching
- You can log in at mystudies.brighton.ac.uk (from inside and outside the university) and you will find information about your course, individual modules and many other things
- Apart from lectures, seminars and labs, this is the main way we communicate with you, mainly using announcements and emails.
- It is important that you check your university email, and MyStudies regularly to get updates about your modules.

The CI401 Module Area

- MyStudies has sections for all the modules running in the university
- If you follow the link to *My modules 2020/21* or the *Studies* tab at the top, all your modules will be listed, including '2020 CI401 – Introduction to Programming'. Click on this to go to the CI401 module area
- The left-hand menu has a list of the main parts of the CI401 module area
- See the Seminar resources for more screenshots (or just play around on the site itself).



Dashboard – 2020 CI401 - Introd

studentcentral.brighton.ac.uk/webapps/blackboard/execute/modulepage/view?course_id=_115415_1&cmp_tab_id=_328756_1&mode=view

Student Preview mode is ON

Settings | Exit Preview

2020 CI401 - Introduction to Programming | Dashboard

2020 CI401 - Introduction to Programming

- Dashboard
- Announcements
- Module Information
- Study Materials
- Reading List
- Assessment and Marks
- Panopto Video

Dashboard

Module Resources

- Module Contacts
- Module Specification
- My Studies Help
- Subject Guide
- LinkedIn Learning
- Accessibility

What's New in your modules, course and school

Edit Notification Settings | Actions

Courses/Organisations (1)

Last Updated: 05 October 2020 21:15

My Announcements

2020 CI401 - Introduction to Programming

› Welcome to CI401

more announcements...

Blackboard

Microsoft Teams

- For information about using Teams, see [Help using Microsoft Teams]
- In teams, select ‘Join or create a team’, and then ‘Join a team with a code’ and give the code *fjfsjqo* . You will join the CI401 team.
- Teams will display the General channel, which we use for announcements
- There is also a Forum channel, for general discussion
- Each week will have its own channel, which is where the lectures, seminars and labs take place, and where the video of the lecture can be found (on the tab bar, with Posts and Files)

BlueJ

- *BlueJ* is the tool we will use to create, develop and test Java programs in this course
- BlueJ is installed on most of the computers in our labs in Cockcroft, and also some of the labs in the Watts building
- You can also install BlueJ on your own computer for free from bluej.org (NB: if you do this, please make sure you download version 4.2.2)
- This week we are focusing mainly on making sure you can run it and do some simple code editing with it.

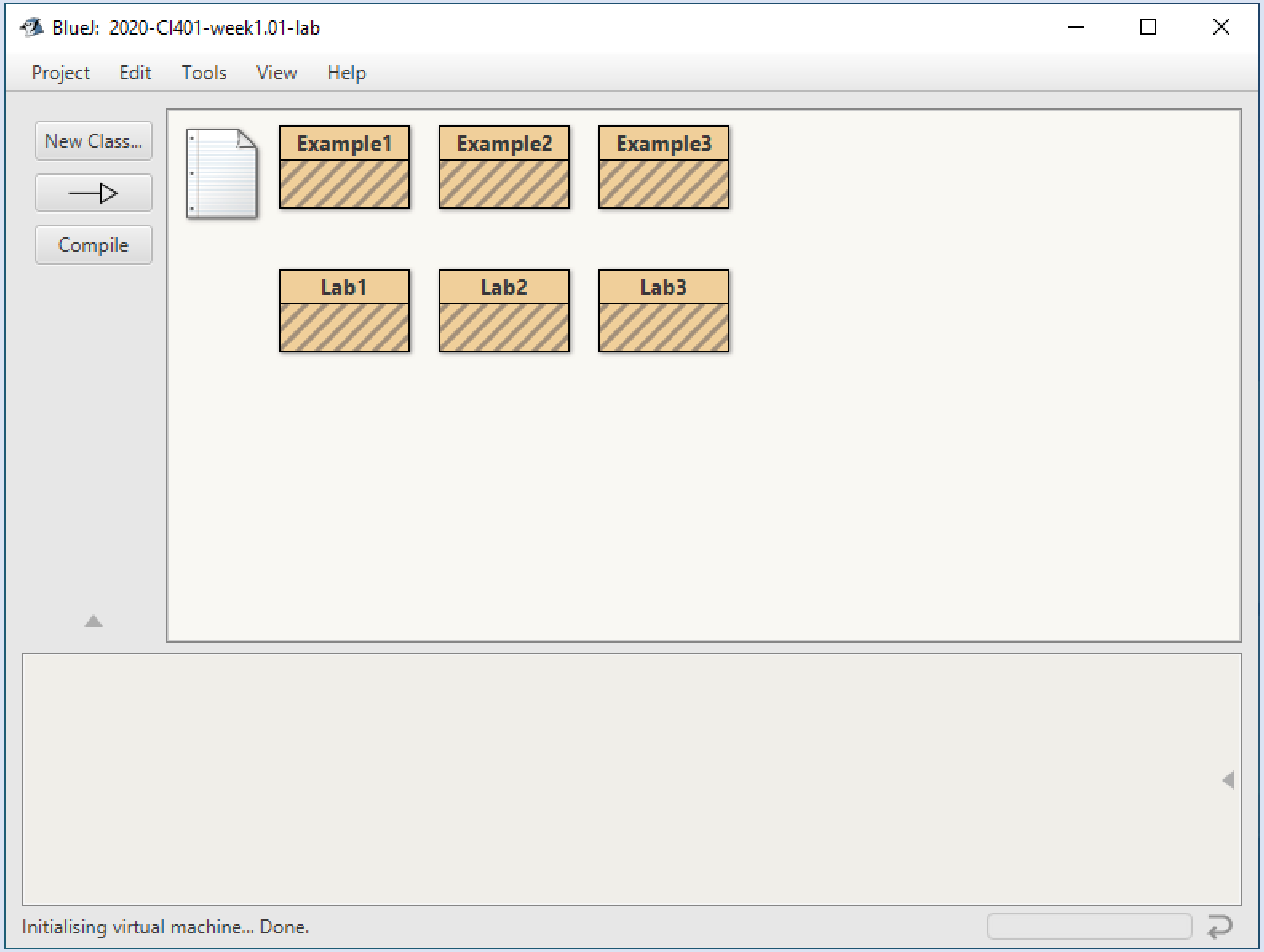
Other IDEs are also available

- BlueJ is an *Integrated Development Environment* or *IDE*
- An IDE is an application which provides all the tools needed for creating new programs – design, coding, testing, documenting etc.
- There are many IDEs available for many different programming languages
- Another very popular IDE is called *Eclipse,* which you may start using in other modules
- You can use it here as well if you want to, but our teaching materials will be mainly based around BlueJ
- Next year, we will focus on using Eclipse, and phase out use of BlueJ

Coding

Getting started with BlueJ

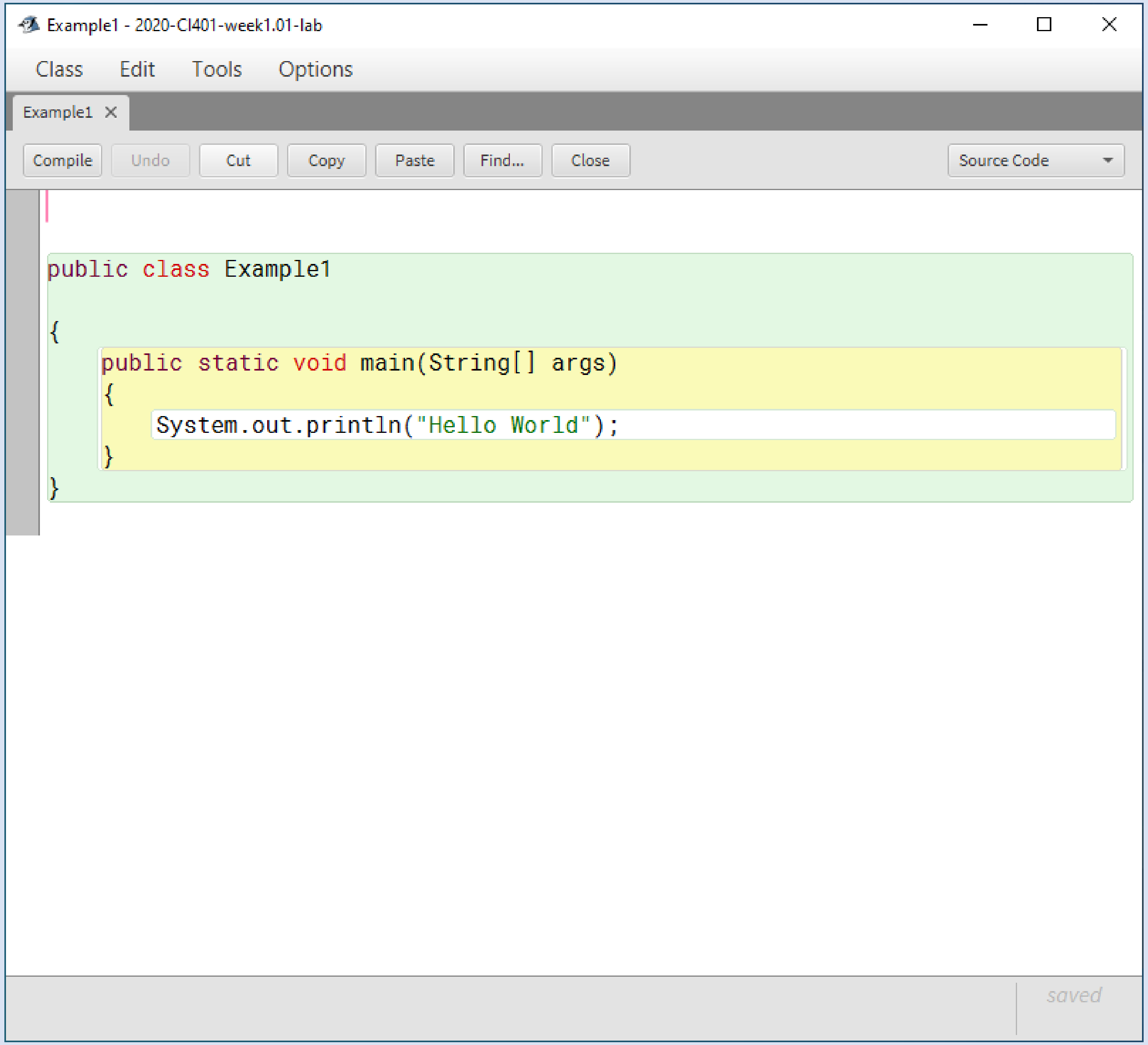
- Run BlueJ from the Windows Start menu like any other app (if you can’t see it, type ‘BlueJ’ into the search bar to find it).
- When BlueJ starts for the first time, it is empty, waiting for you to create or install a ‘project’ – usually a Java program.
- You will find a small BlueJ project for this week’s lab sessions in a ‘jar’ (Java Archive) file in the module area on StudentCentral. It is called **week1.01-lab.jar** and you need to download it to your local machine.
- Once you have it, you can create a project, by selecting **Project->Open ZIP/JAR…** and browse to the jar file to install.
- And then you are ready to code!



Initialising virtual machine... Done.

Example 1 – Hello World

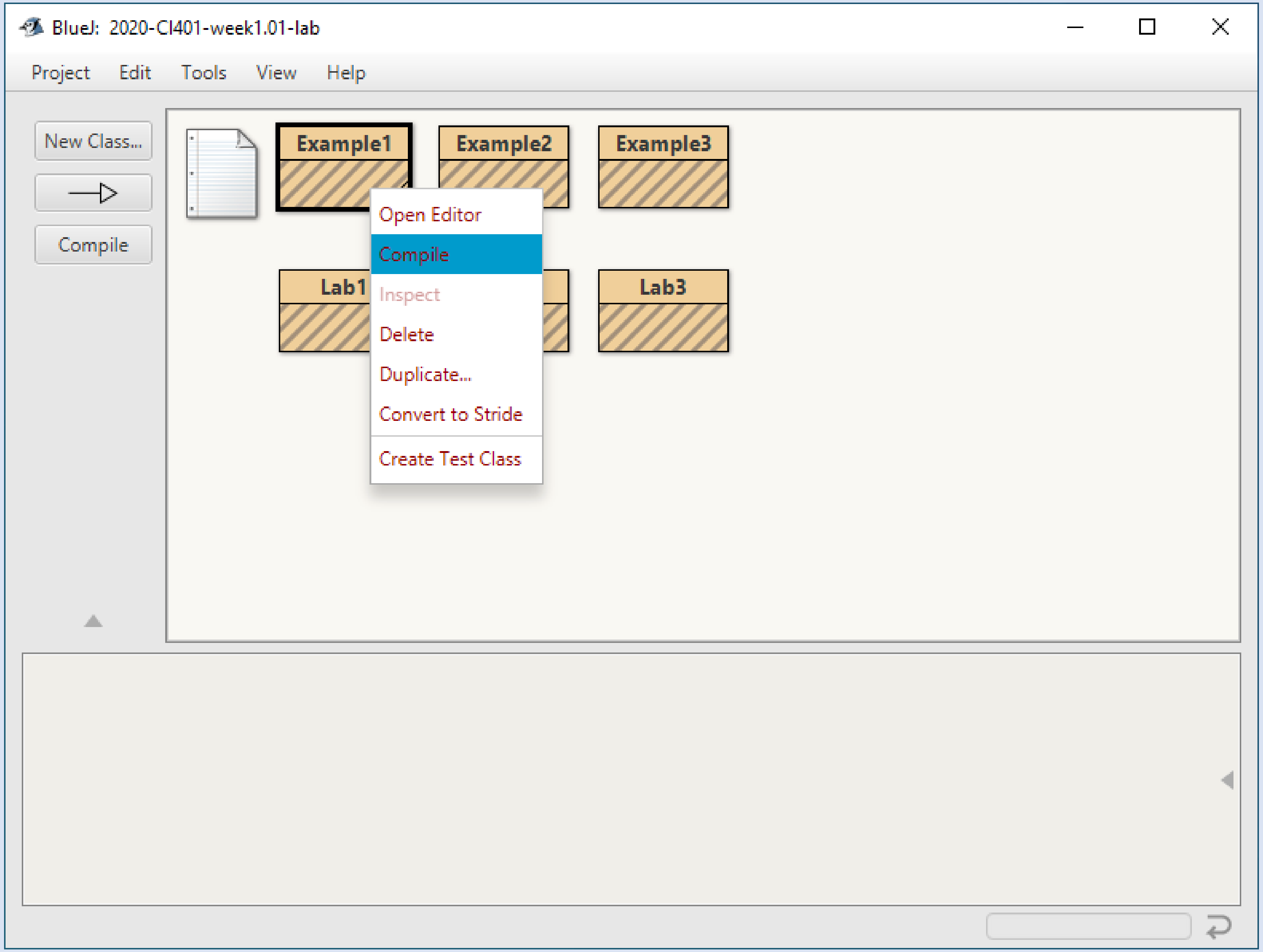
- Open Example1 (in your BlueJ project) by double-clicking on it
- What you get is some coloured boxes with writing in them (most of which doesn't make much sense)
- This is a *Java program* – that is a computer program written in the programming language Java.
- It is just about the simplest Java program you can write
- It just tells the computer to write "Hello World" on the screen
- This is traditionally the first program you see in any programming language course

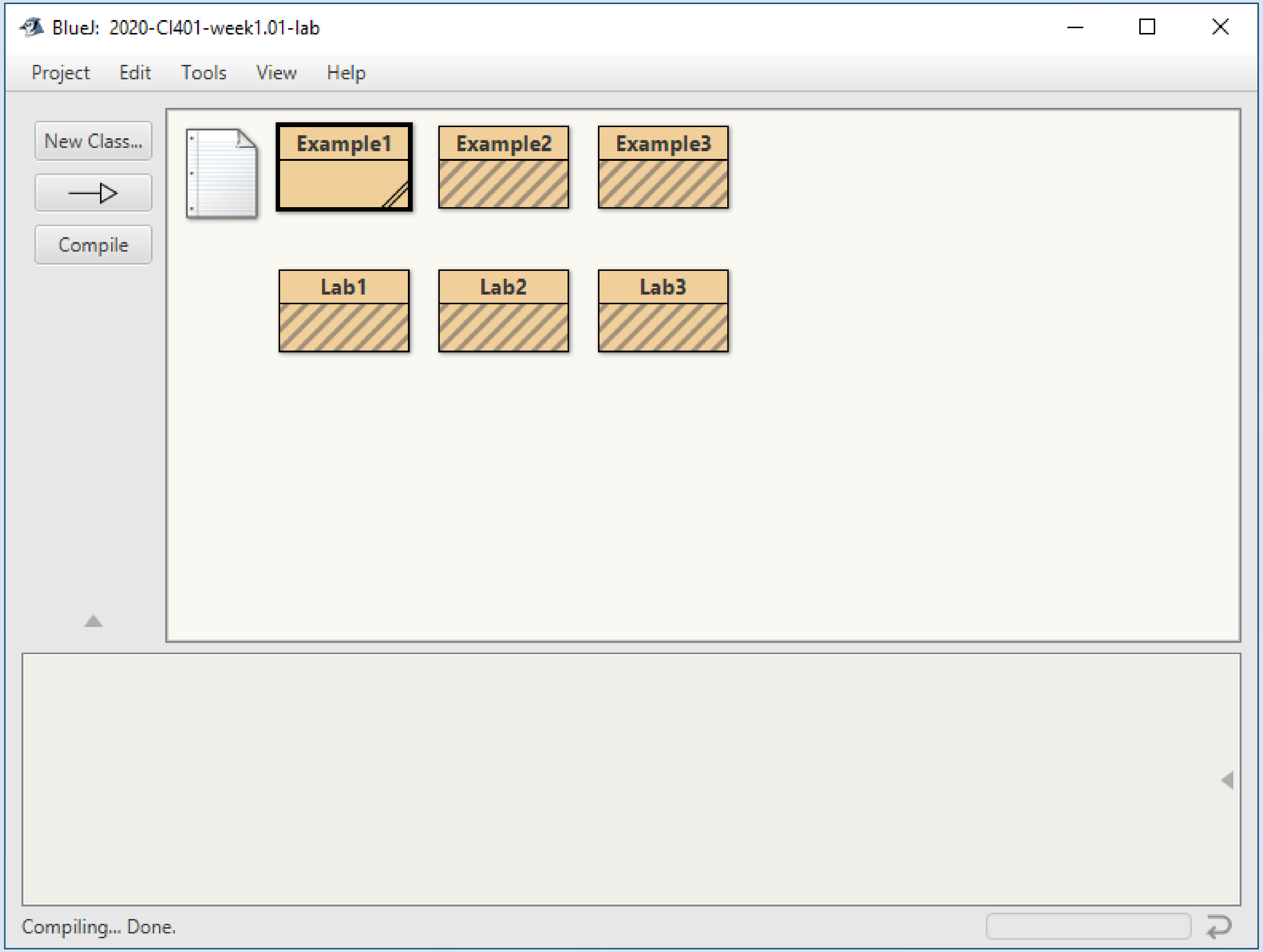


```
public class Example1

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

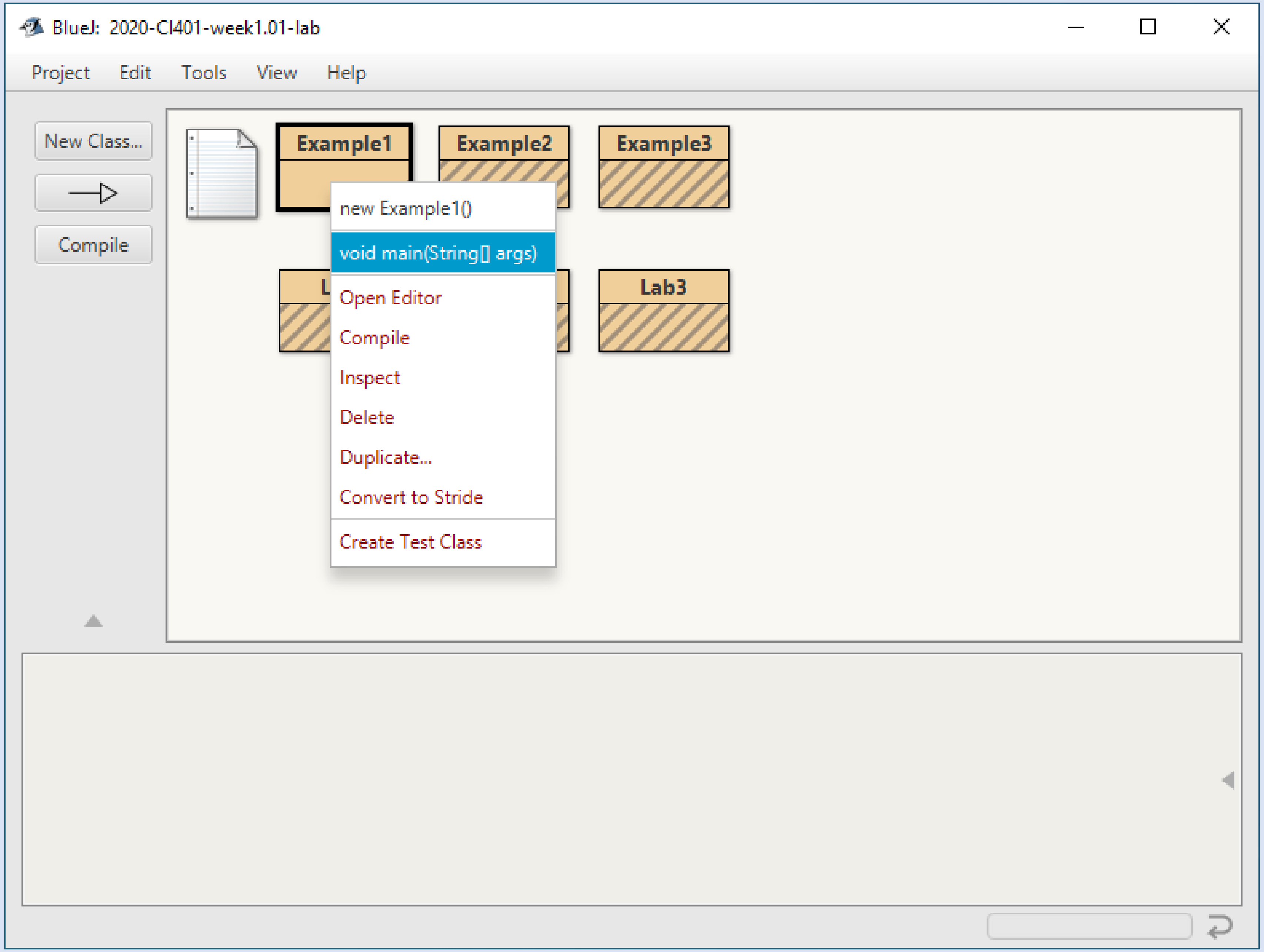
Making a computer do something

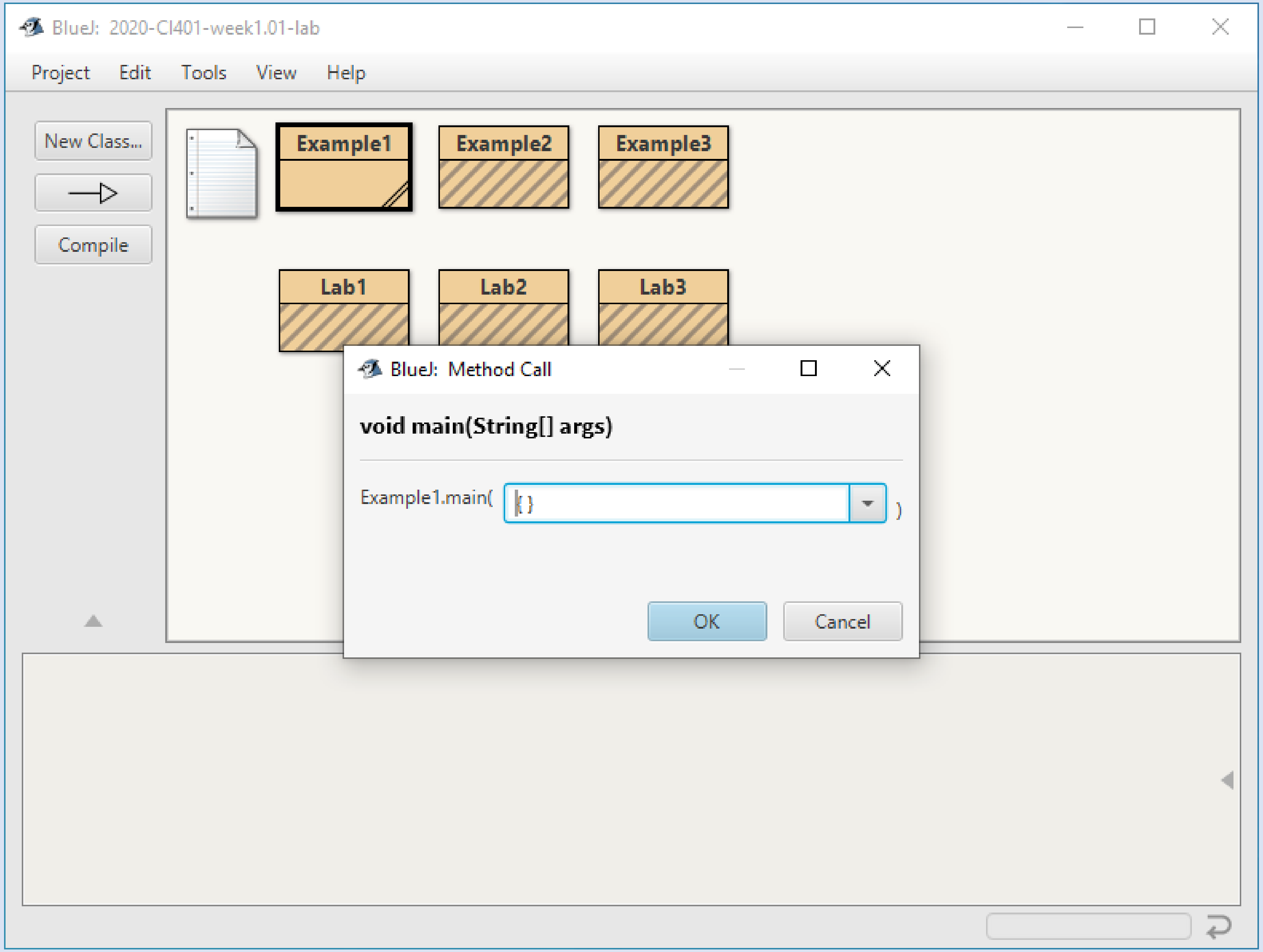
- We will look at the program closely in a minute. First let’s look at how this program can be used to make a computer do something.
- The program is basically a set of instructions. When we get the computer to carry out those instructions, we say that the computer is *running the program*
- To run a Java program, there are two steps.
  - First you have to *compile* it (convert it from Java into a form the computer understands more directly)
  - Then you *execute* it (tell the computer to run the compiled version).
- Let’s do that first, to check it does what it should, before looking at the code more closely


BlueJ: 2020-CI401-week1.01-lab
Project
Edit
Tools
View
Help
New Class...
Compile
Example1
Example2
Example3
Lab1
Lab3
Open Editor
Compile
Inspect
Delete
Duplicate...
Convert to Stride
Create Test Class



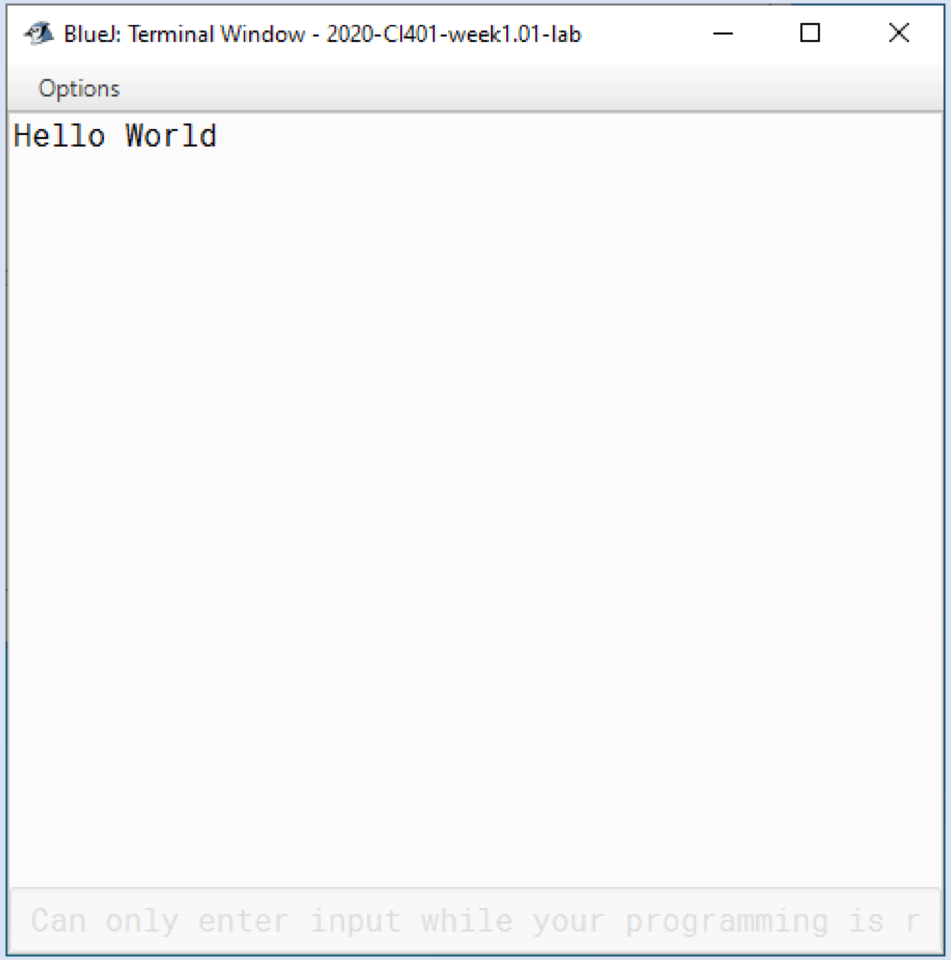
Compiling... Done.


BlueJ: 2020-CI401-week1.01-lab
Project
Edit
Tools
View
Help
New Class...
Compile
Example1
Example2
Example3
Lab3
new Example1()
void main(String[] args)
Open Editor
Compile
Inspect
Delete
Duplicate...
Convert to Stride
Create Test Class



void main(String[] args)

Example1.main({ })


BlueJ: Terminal Window - 2020-CI401-week1.01-lab
Options

```
Hello World
```

Can only enter input while your programming is r

Running a program in BlueJ

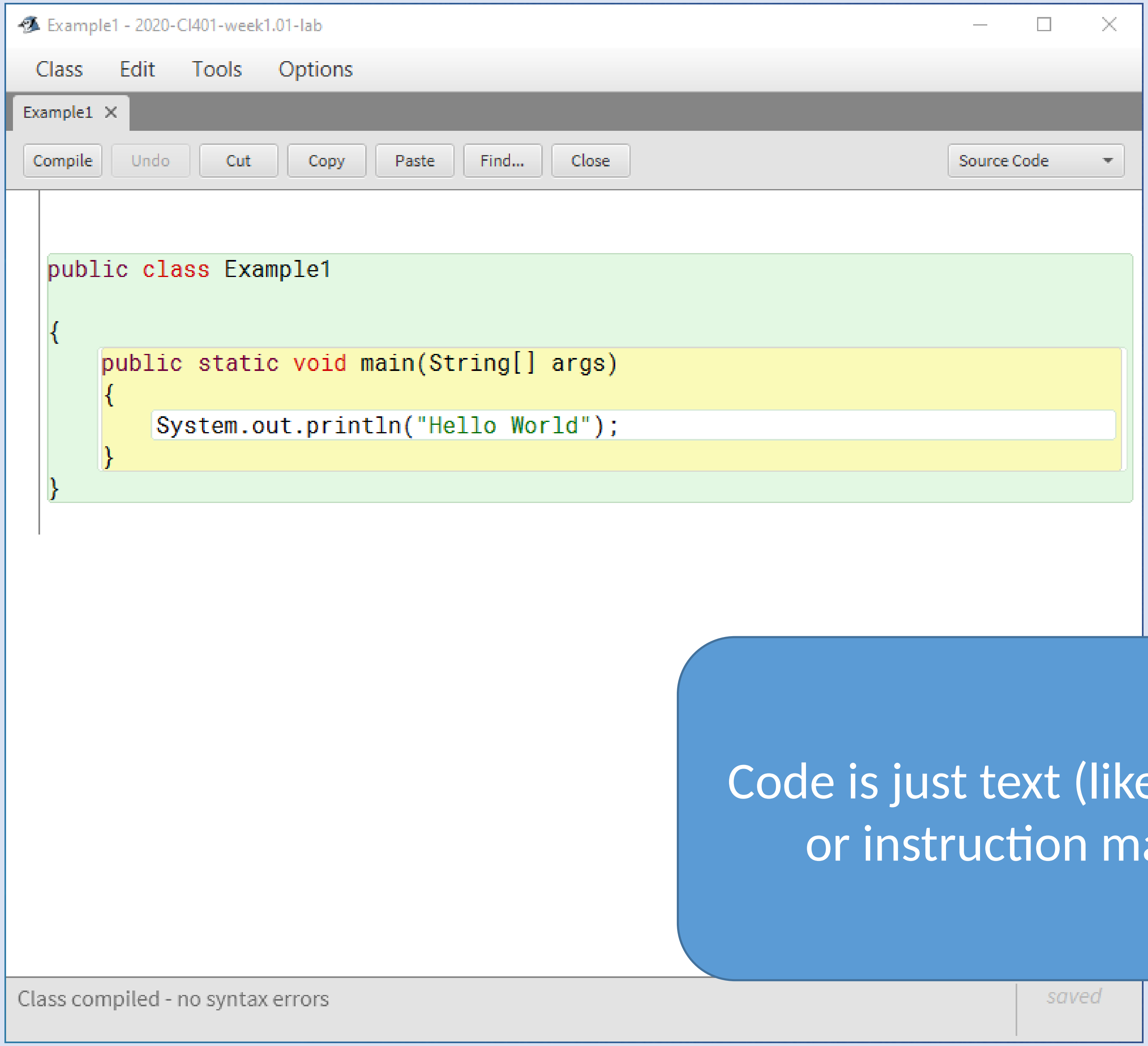
- Each of the boxes on the BlueJ screen is a Java program (or part of one)
- To run Example1 we left-click on it and select the *Compile* option
- When we left-click again there are two new options and we choose the rather mysterious *void main(String[] args)* to make BlueJ execute the compiled program
- When we do that, BlueJ pops up a little box in which we just click ‘OK’
- Finally BlueJ pops up another box saying *Hello World*

So what just happened?

- We started with a program that (I claimed) instructed the computer to write *Hello World* on the screen, and indeed it did, in that final window that popped up.
- Everything else – the menus, pop-up windows, mysterious incantations – was just BlueJ setting things up to run the program, and providing a way for the program to show something ‘on the screen’.
- All the program itself did, was that little bit of writing “Hello World”
- Change the program to say something else, and you will see that’s the only thing that changes.

Java programs

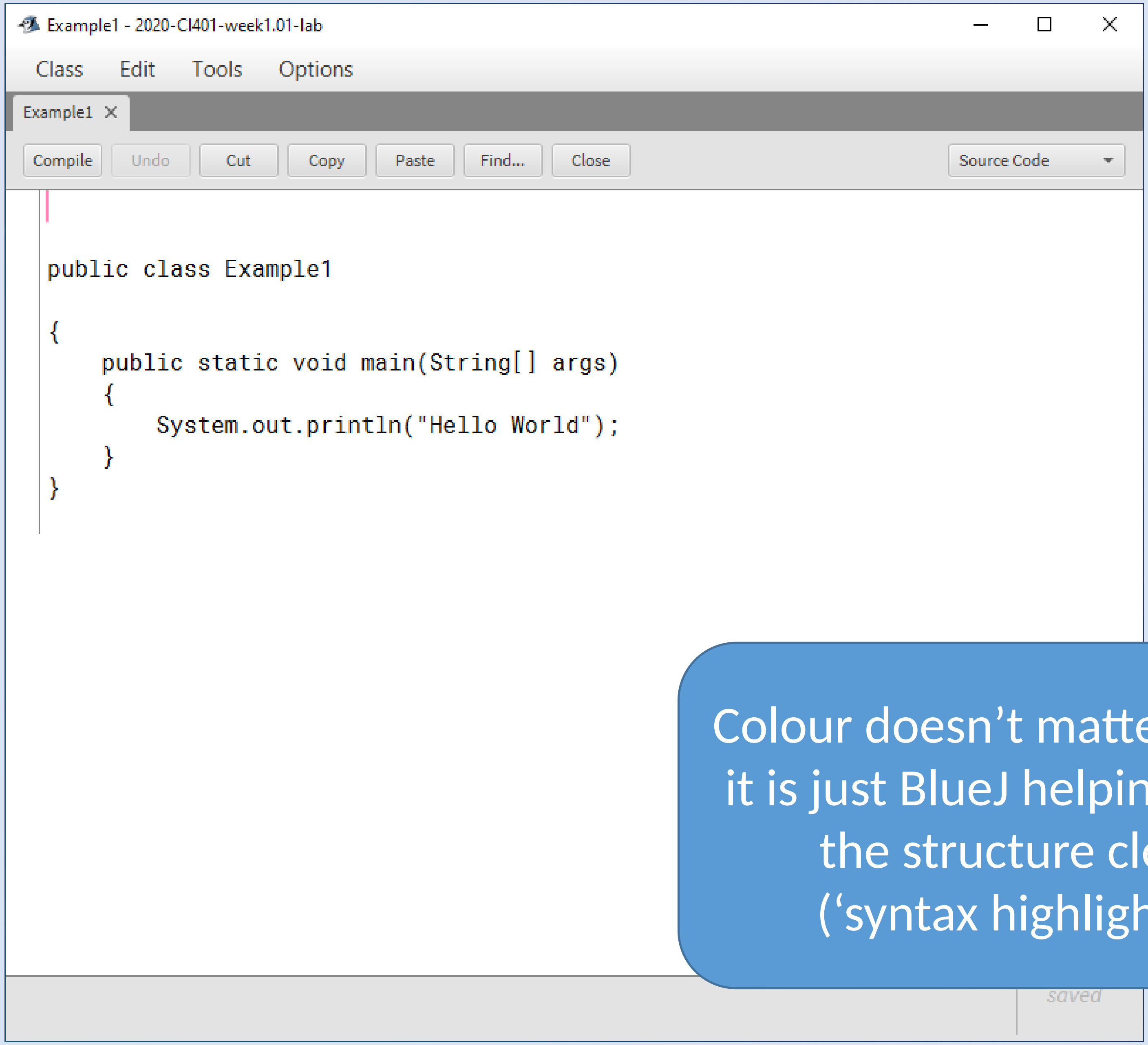
- A program is just a set of instructions for the computer
- But the instructions are specified in a very precise way
- You have to be very careful about punctuation and spelling
- Layout is less important (to the computer)
- Colour is just added by BlueJ to make it easier to see what you are doing.



```
public class Example1

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

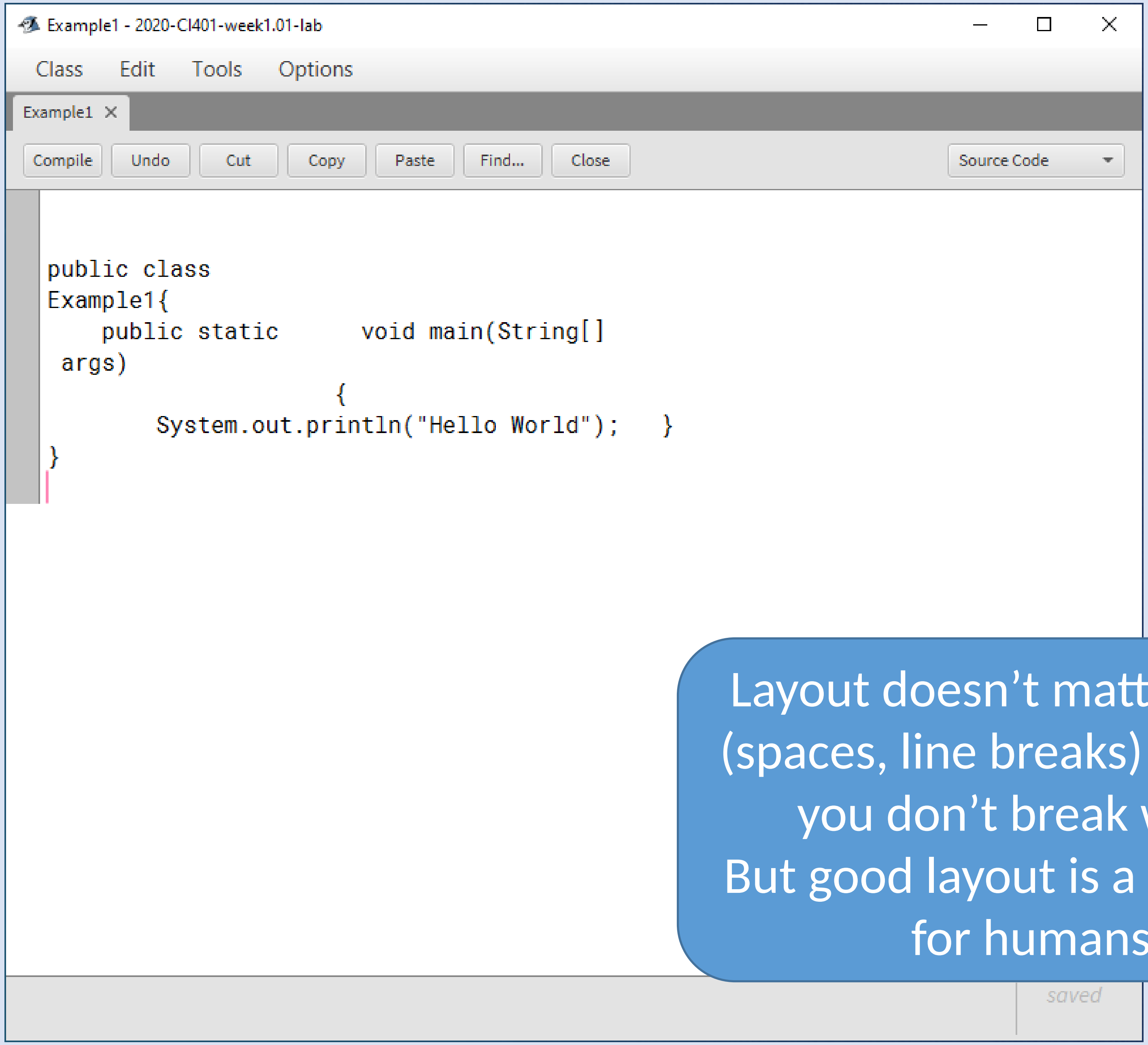
Class compiled - no syntax errors

Code is just text (like a recipe, or instruction manual)



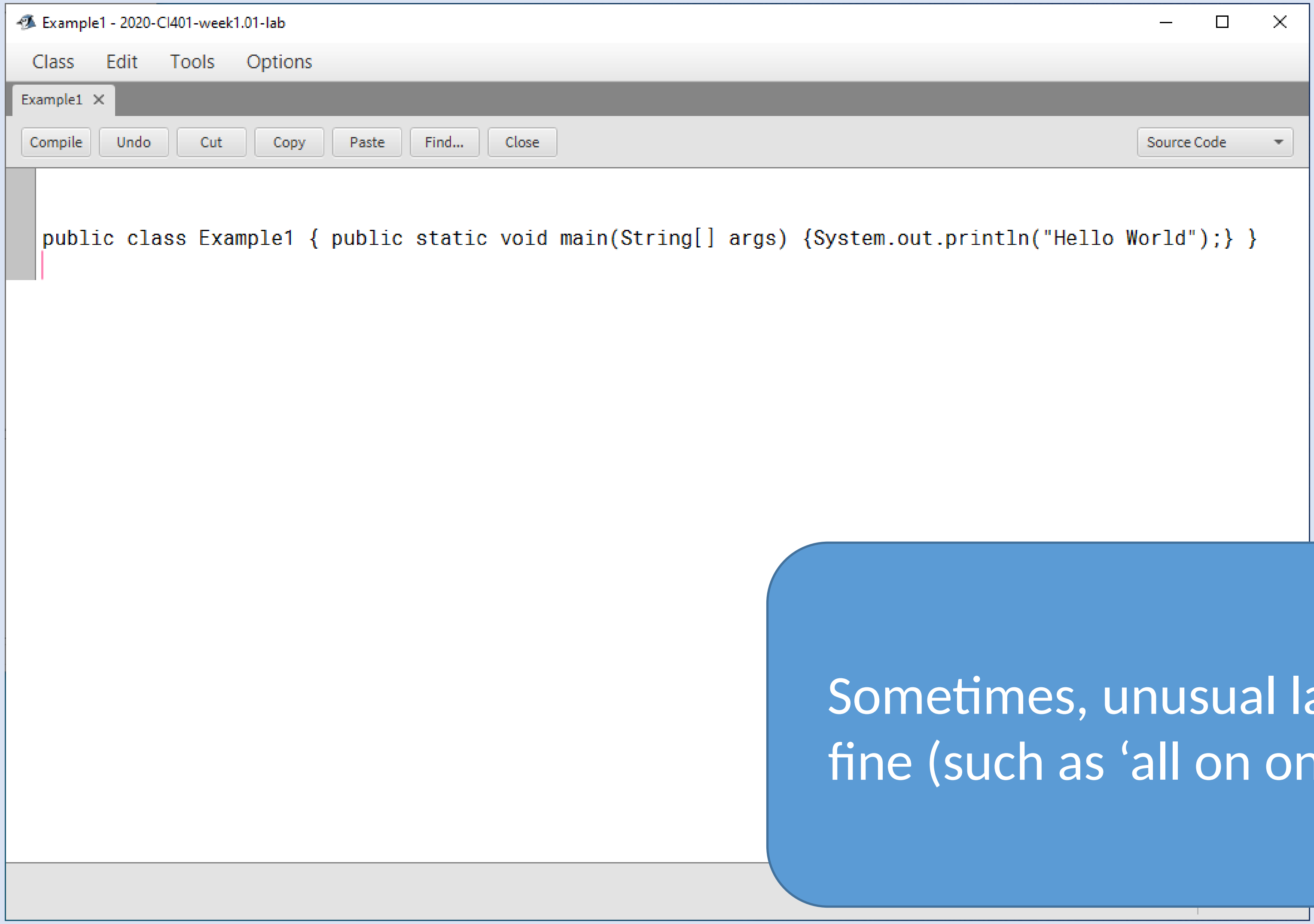
```
public class Example1

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

Colour doesn’t matter to Java – it is just BlueJ helping to make the structure clearer (‘syntax highlighting’)



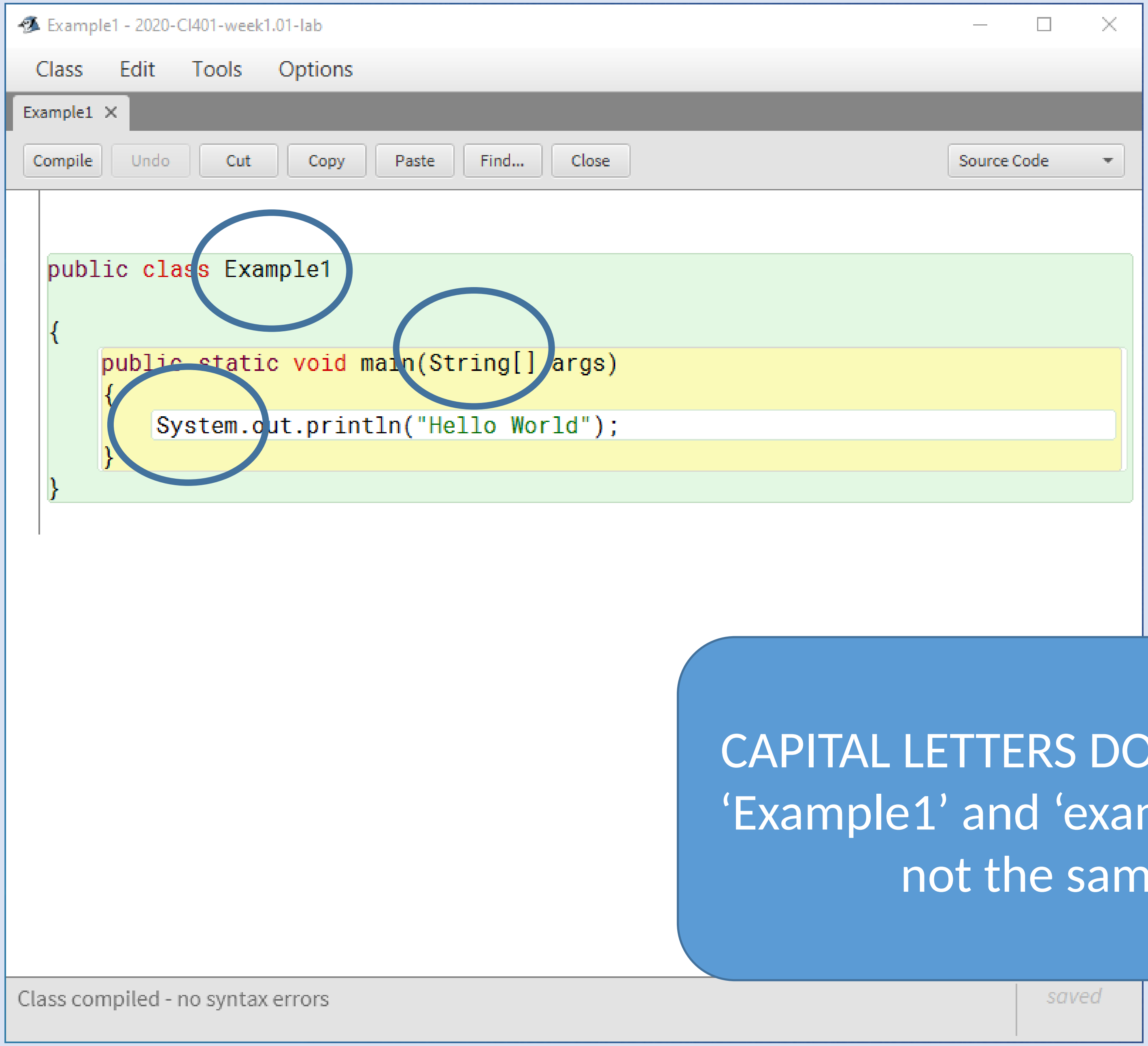
```
public class
Example1{
    public static      void main(String[]
 args)
                     {
        System.out.println("Hello World");   }
}
```

Layout doesn't matter to Java (spaces, line breaks) as long as you don't break words.
But good layout is a good idea for humans!



```
public class Example1 { public static void main(String[] args) {System.out.println("Hello World");} }
```

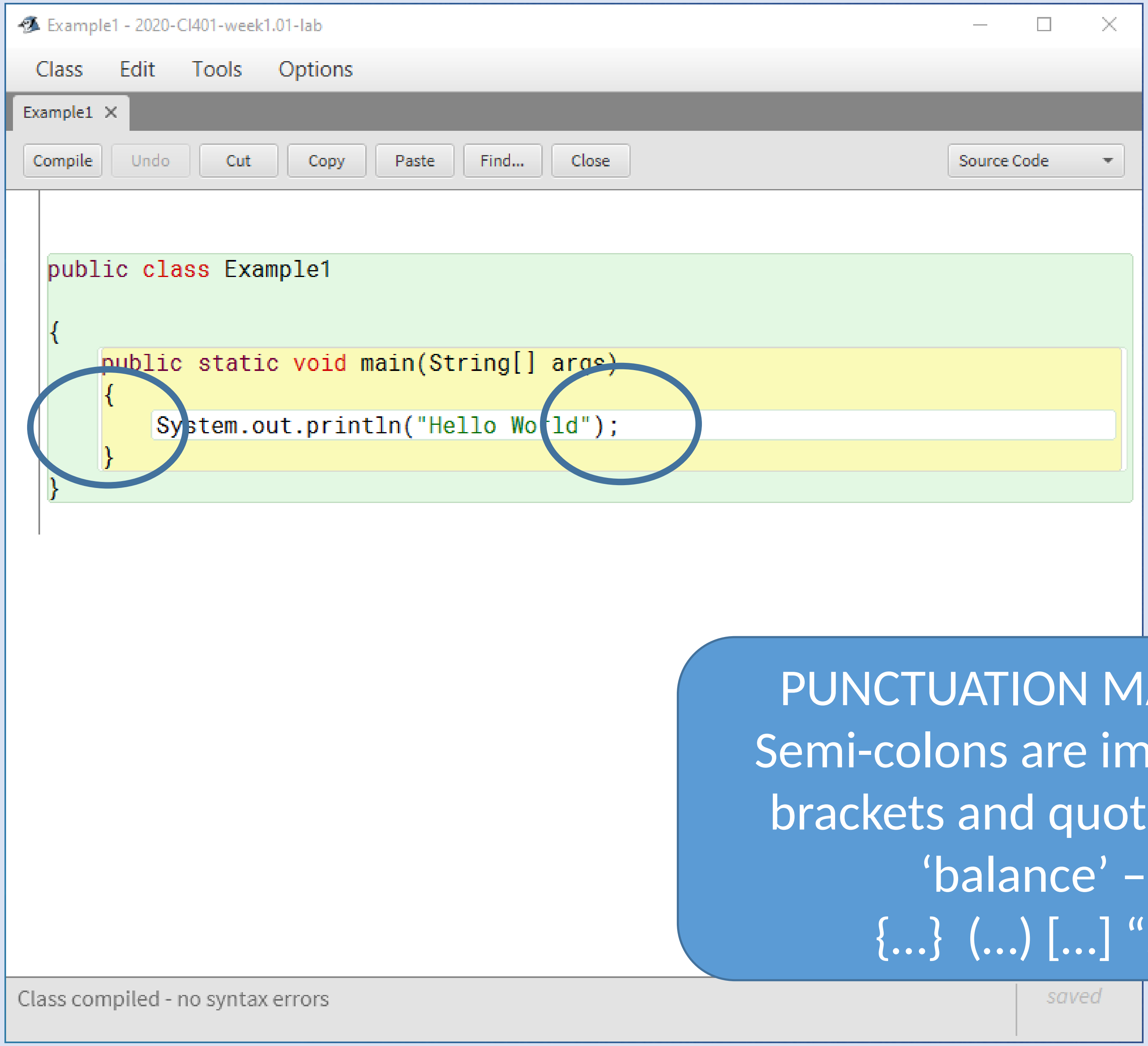
Sometimes, unusual layout is fine (such as 'all on one line')



```
public class Example1

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

Class compiled - no syntax errors

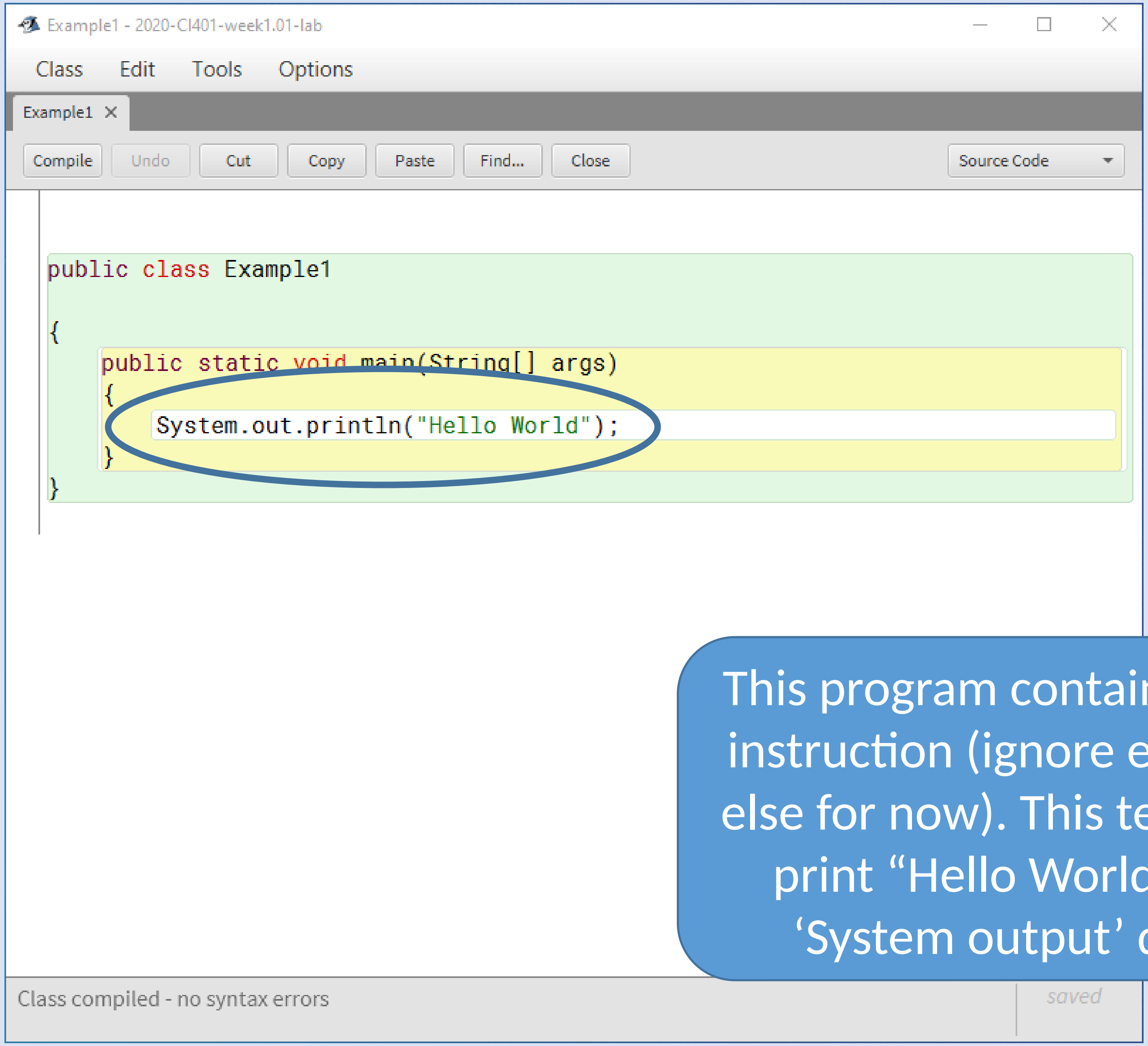
CAPITAL LETTERS DO MATTER! ‘Example1’ and ‘example1’ are not the same



```
public class Example1

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

Class compiled - no syntax errors

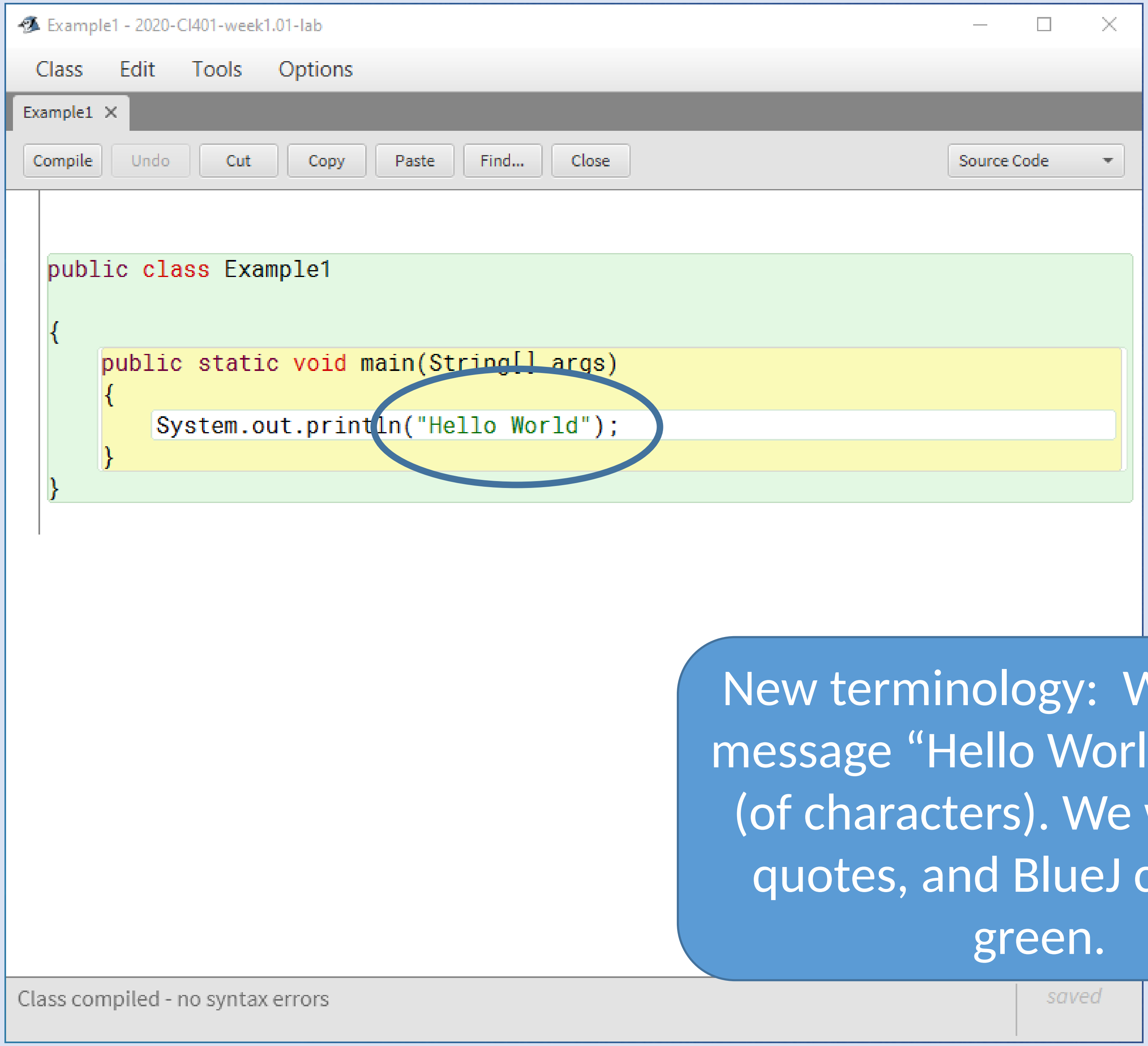
PUNCTUATION MATTERS
Semi-colons are important, brackets and quotes must 'balance' –
{…} (…) […] "…"



```
public class Example1

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

Class compiled - no syntax errors

This program contains a single instruction (ignore everything else for now). This tells Java to print "Hello World" on its 'System output' device.

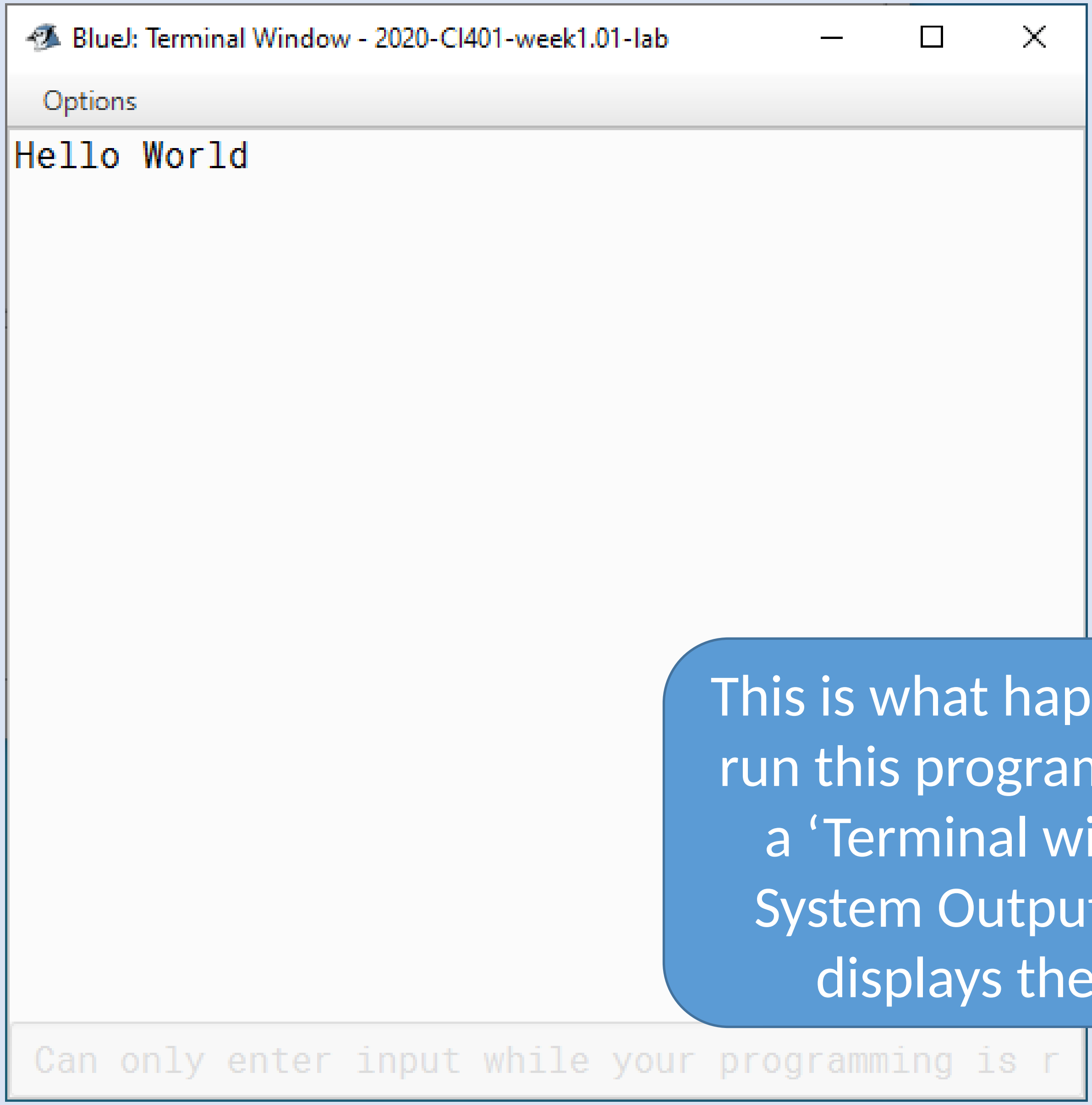


```
public class Example1

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

Class compiled - no syntax errors

saved

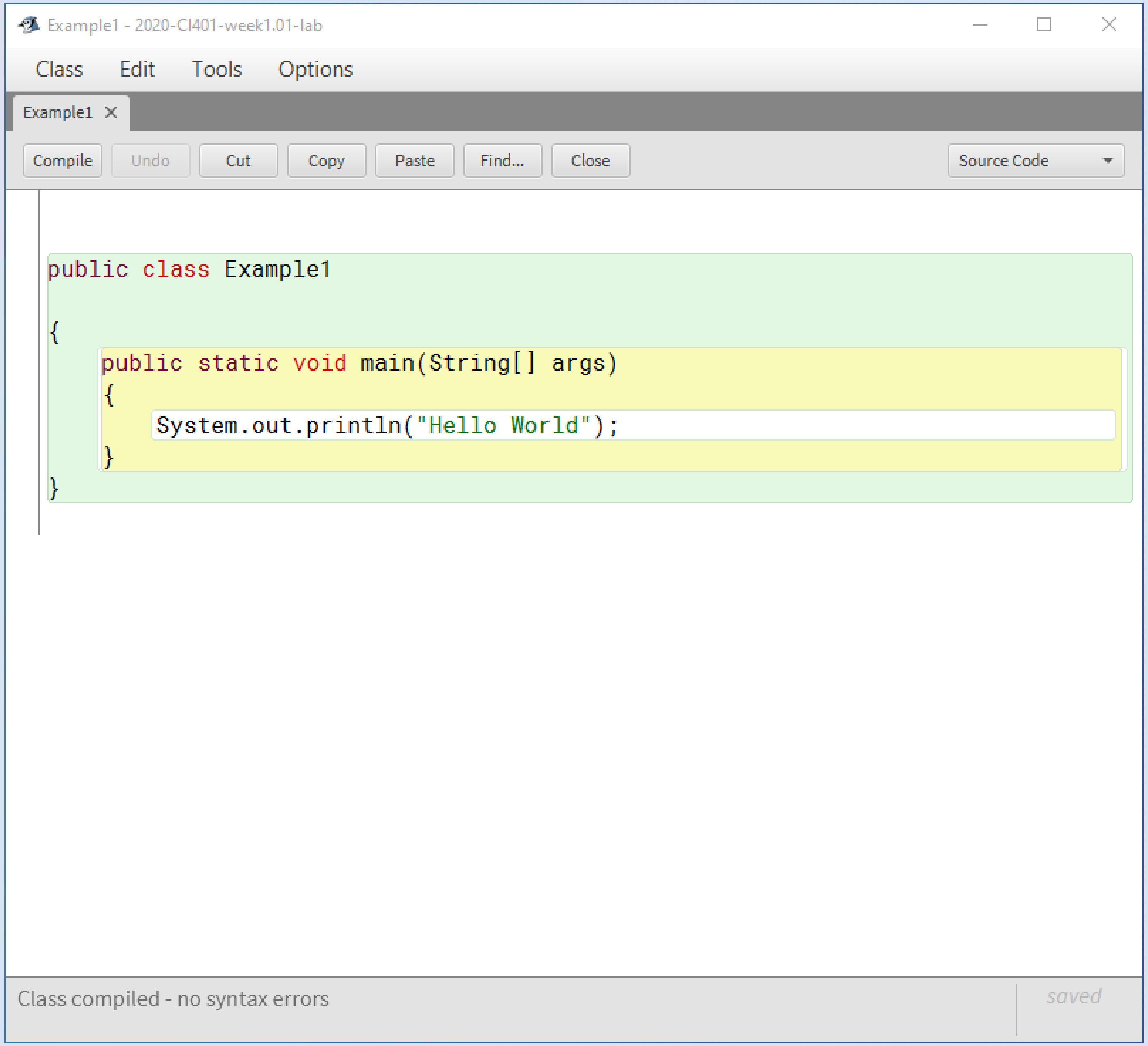
New terminology: We call the message “Hello World” a String (of characters). We write it in quotes, and BlueJ colours it green.



```
Hello World
```

Can only enter input while your programming is r

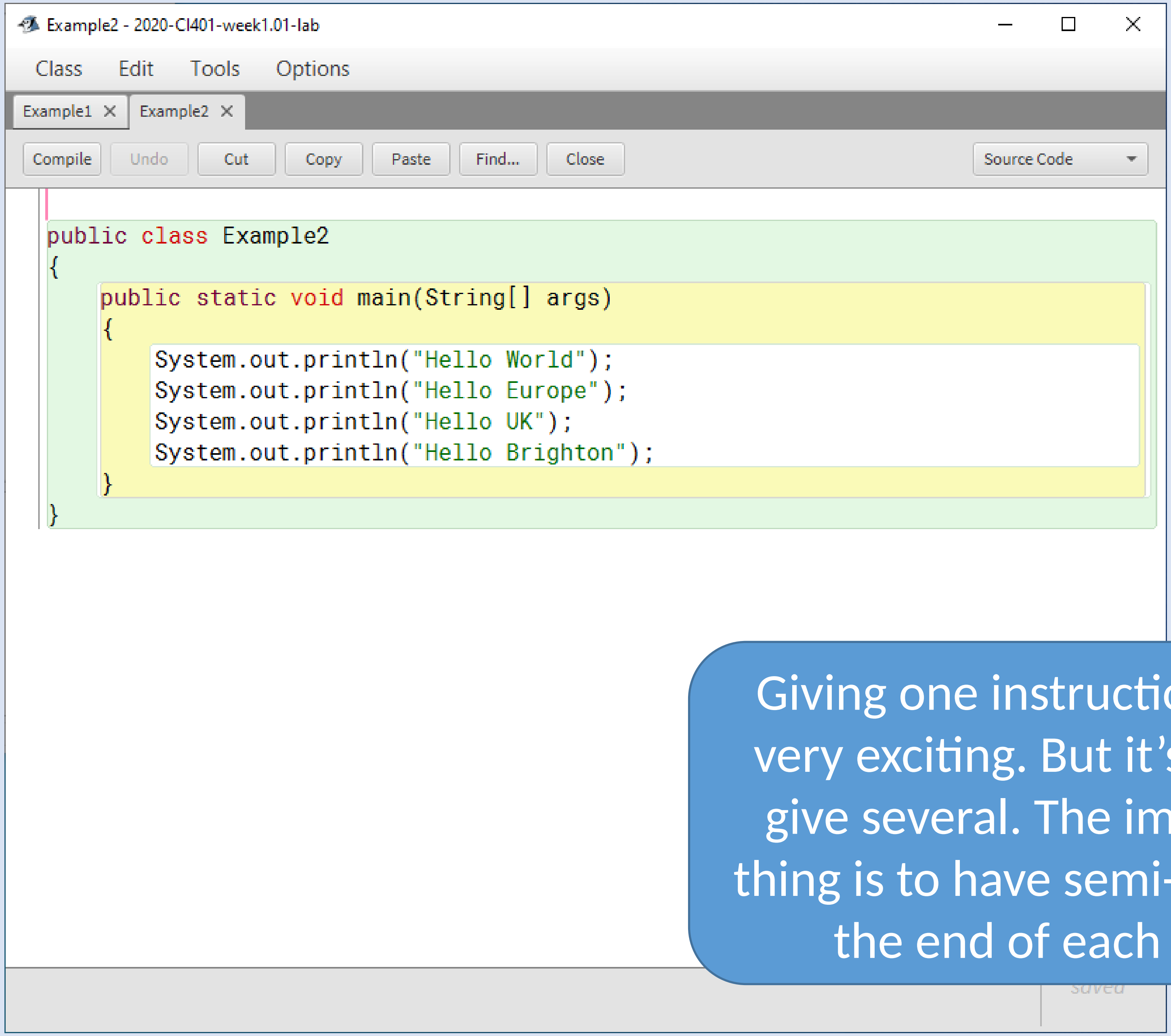
This is what happens when we run this program. BlueJ opens a ‘Terminal window’ as its System Output device, and displays the message.



```
public class Example1

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

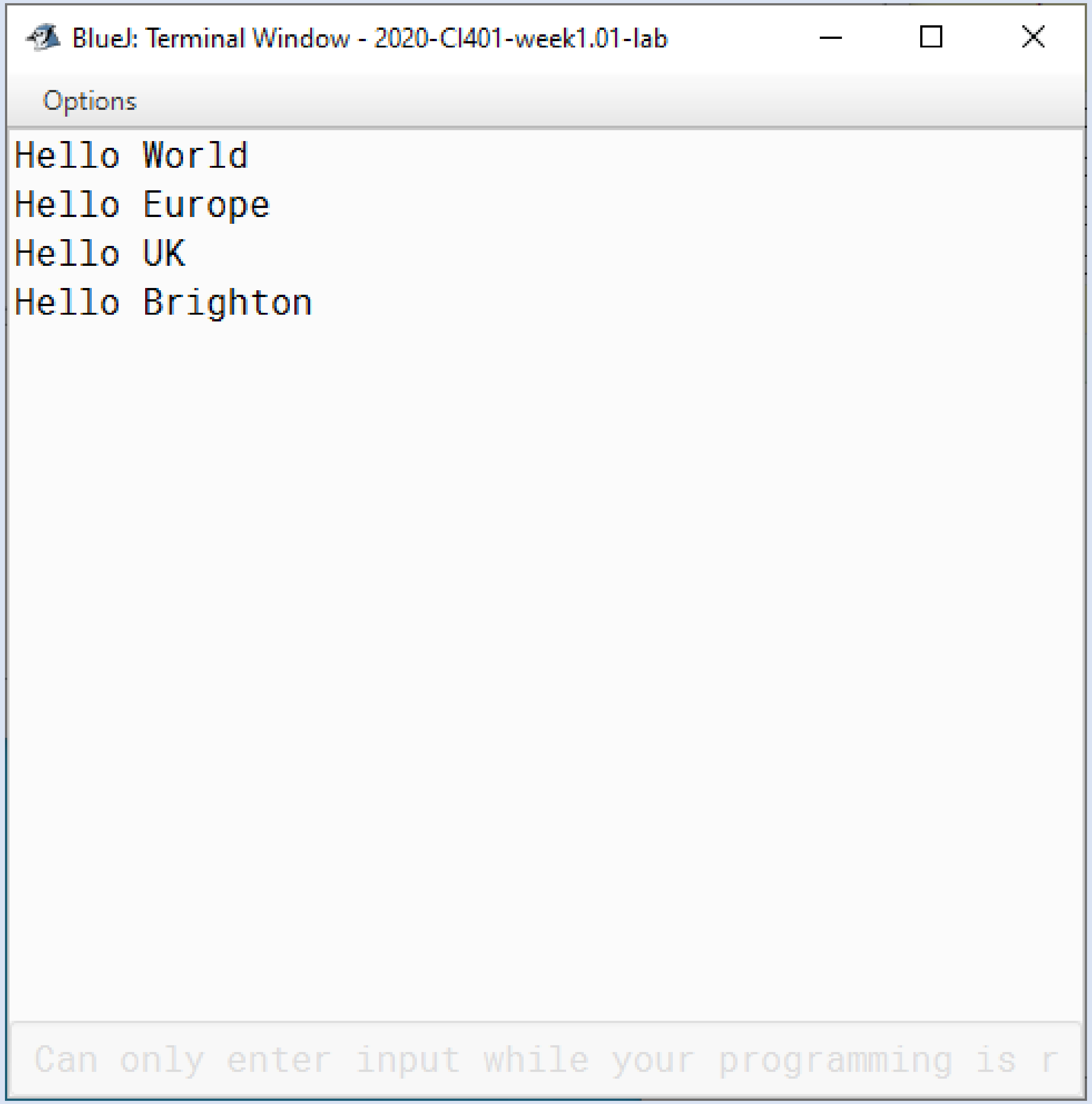
Class compiled - no syntax errors

saved


Example2 - 2020-CI401-week1.01-lab
Class Edit Tools Options
Example1 Example2
Compile Undo Cut Copy Paste Find... Close Source Code
saved

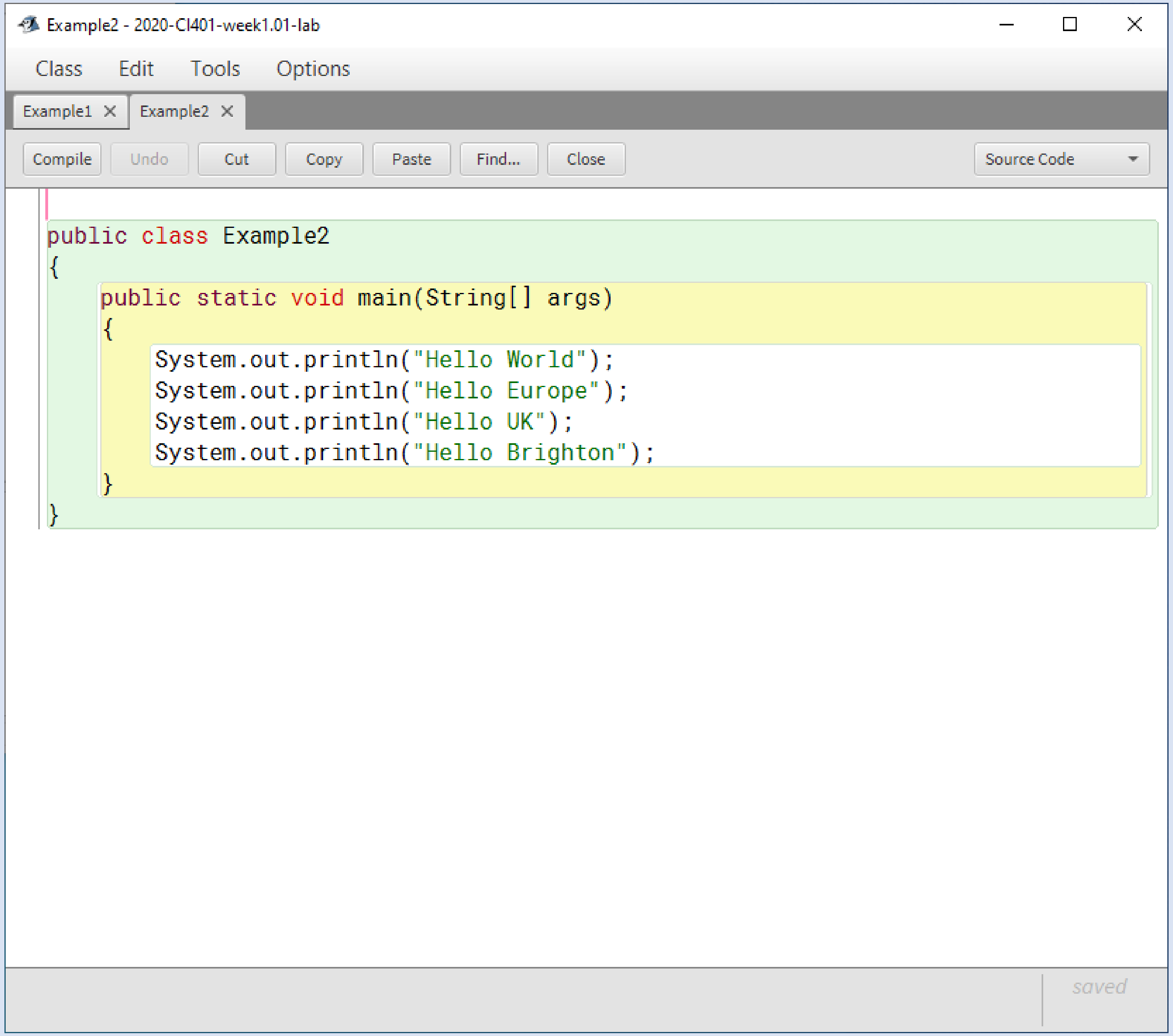
```
public class Example2
{
    public static void main(String[] args)
    {
        System.out.println("Hello World");
        System.out.println("Hello Europe");
        System.out.println("Hello UK");
        System.out.println("Hello Brighton");
    }
}
```

Giving one instruction is not very exciting. But it's easy to give several. The important thing is to have semi-colons at the end of each one.

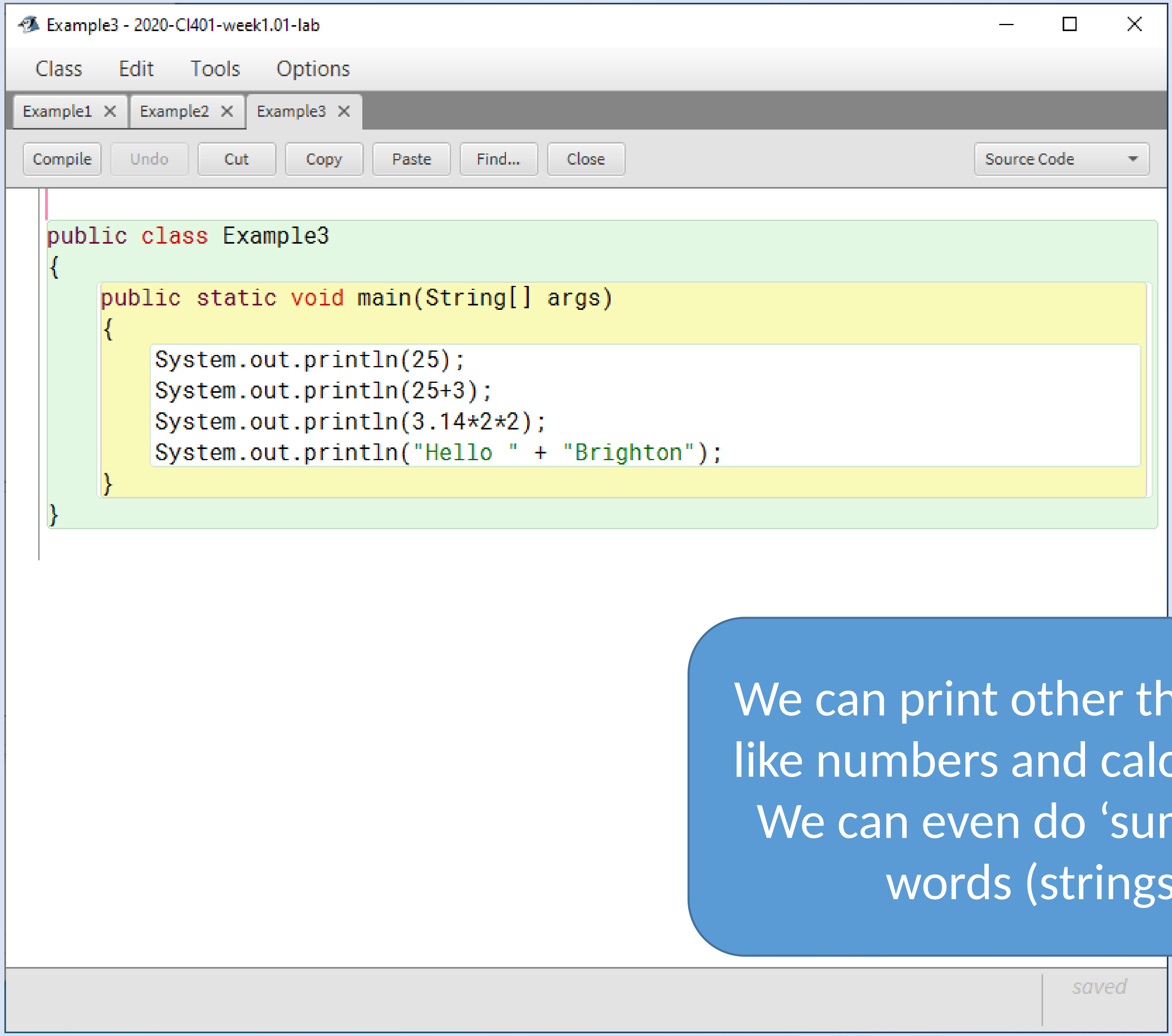


```
Hello World
Hello Europe
Hello UK
Hello Brighton
```

```
Can only enter input while your programming is r
```

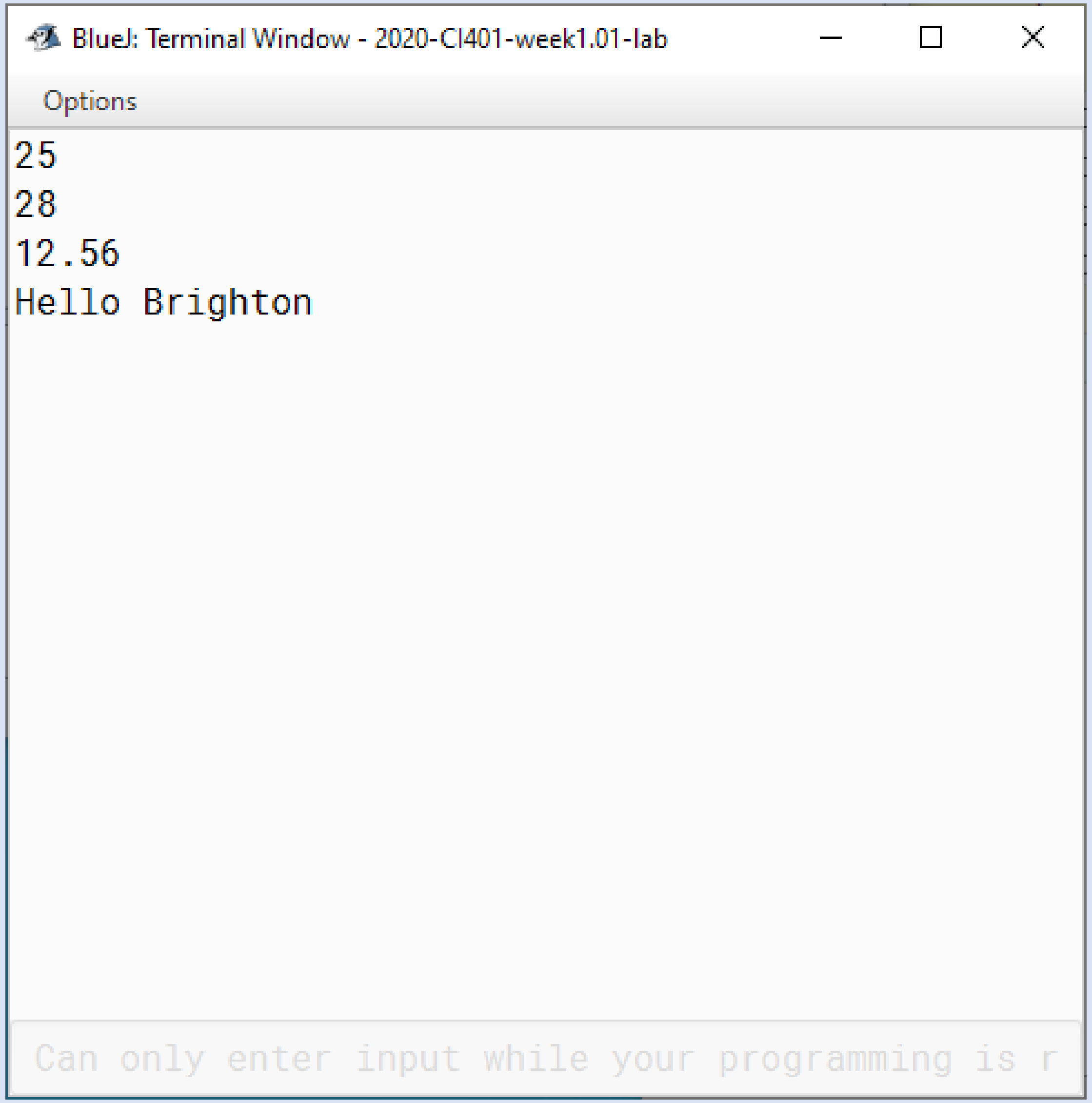


```
public class Example2
{
    public static void main(String[] args)
    {
        System.out.println("Hello World");
        System.out.println("Hello Europe");
        System.out.println("Hello UK");
        System.out.println("Hello Brighton");
    }
}
```


Example3 - 2020-CI401-week1.01-lab
Class Edit Tools Options
Example1 Example2 Example3
Compile Undo Cut Copy Paste Find... Close
Source Code
saved

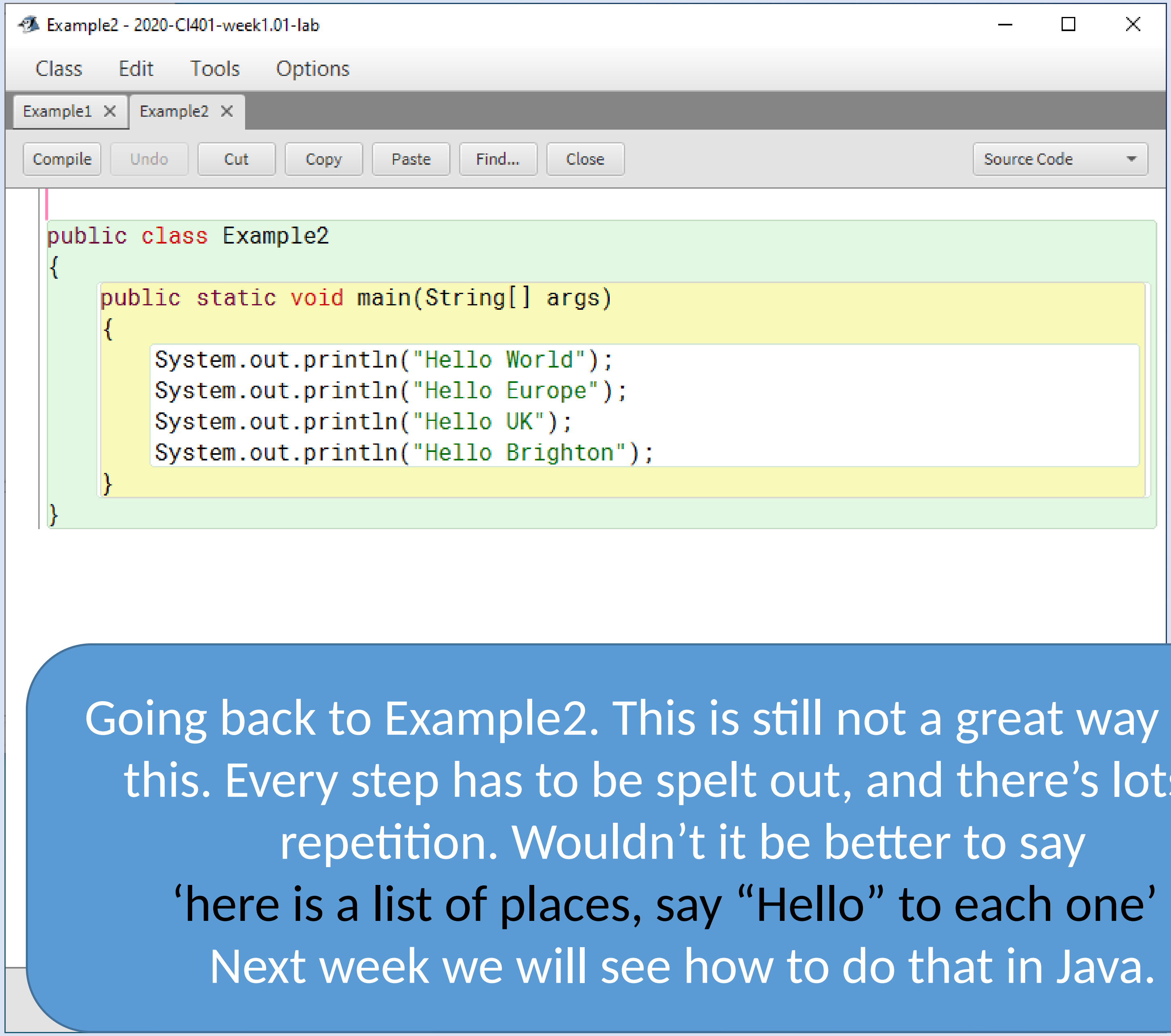
```
public class Example3
{
    public static void main(String[] args)
    {
        System.out.println(25);
        System.out.println(25+3);
        System.out.println(3.14*2*2);
        System.out.println("Hello " + "Brighton");
    }
}
```

We can print other things too, like numbers and calculations. We can even do 'sums' with words (strings).



```
25
28
12.56
Hello Brighton
```

```
Can only enter input while your programming is r
```


Example2 - 2020-CI401-week1.01-lab
Class Edit Tools Options
Example1 Example2
Compile Undo Cut Copy Paste Find... Close
Source Code

```
public class Example2
{
    public static void main(String[] args)
    {
        System.out.println("Hello World");
        System.out.println("Hello Europe");
        System.out.println("Hello UK");
        System.out.println("Hello Brighton");
    }
}
```

Going back to Example2. This is still not a great way to do this. Every step has to be spelt out, and there's lots of repetition. Wouldn't it be better to say
'here is a list of places, say "Hello" to each one' ?
Next week we will see how to do that in Java.

Week 1.01 Labs

Attending labs

- Labs sessions will run on campus and on line simultaneously
- Your personal timetable will show you which lab you are assigned to
- (We don't mind a bit of moving around, but if a lab is too busy, we may need to get stricter)
- You can choose whether to come onto campus for a lab, or attend online through teams
- Labs are for doing individual work, and getting help from the tutor if needed
- You can work on your own laptop, or a lab machine

Lab exercises – MyStudies

- Log in on a lab computer
- Access mystudies.brighton.ac.uk, and log in
- Find your personal timetable, and check that you are in the right lab session 🏝
- Find the CI401 module (from My Modules 2019/20, or the Studies tab)
- Find the Announcements section – this is the way we usually communicate with you (apart from lectures and labs)
- Find the Study Materials section and click through week1.01 to find the slides (week1.01-lecture.pptx) and lab file (week1.01-lab.jar)

Lab exercises – BlueJ

- Create a folder for this week's work on your S: drive (or on your personal computer) e.g. at S:\CI401\week1.01
- Download BlueJ project week1.01-lab.jar from MyStudies into this folder
- Open BlueJ on your computer and create a new project from the jar file in your new folder (see the lecture slides to find out how)
- BlueJ will show you a folder full of Example files and Lab exercises

Lab exercises – coding

- Open each of the example files (double click them) and look at the code. Try to understand what each one does (look at the lecture slides too), and then compile and run it to see if you were right
- Open each of the lab files and follow the instructions at the top to edit the code to do something new. Then compile and run it to see if it works.
- Remember to save your work to your S: drive before finishing.

Taking work home

- This year, it is more important that usual to be able to work at home as well as on campus
- If you use a lab computer for the exercises, you save your work on the S: drive – **but you can’t access the S: drive from home**
- If you want to access your work at home as well, copy the week1.01 folder from your S: drive to your O: drive
- Then, at home, you can open your university onedrive account (in a web browser), and download your work.
- Remember to copy work back to onedrive at home before coming on campus

Lab exercises – we are here to help!

- **If you get stuck, ask for help!**
- **Even if you don’t get stuck, talk to us!**