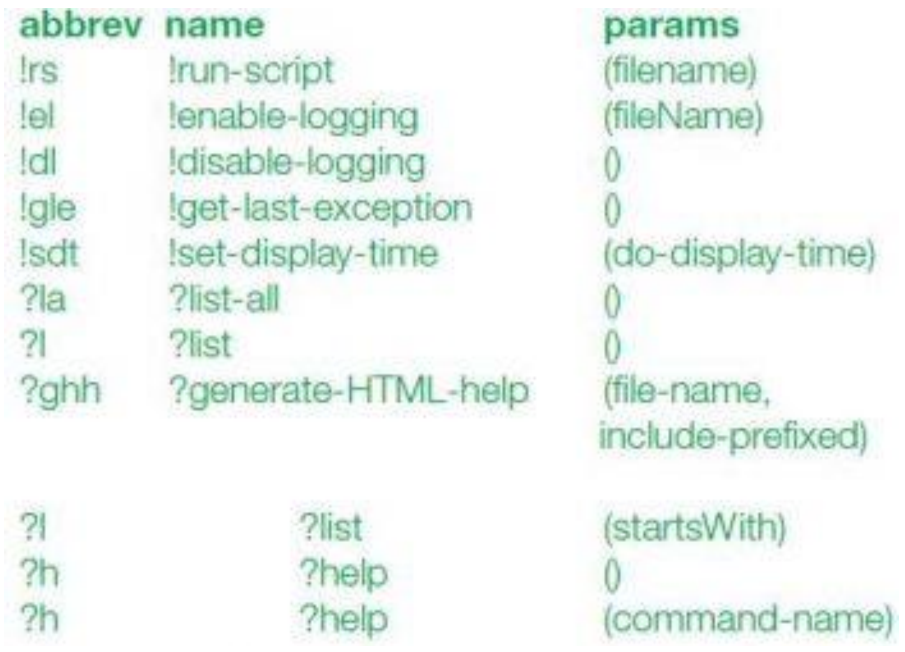
Frank Walsh

Cliche

What’s Cliche

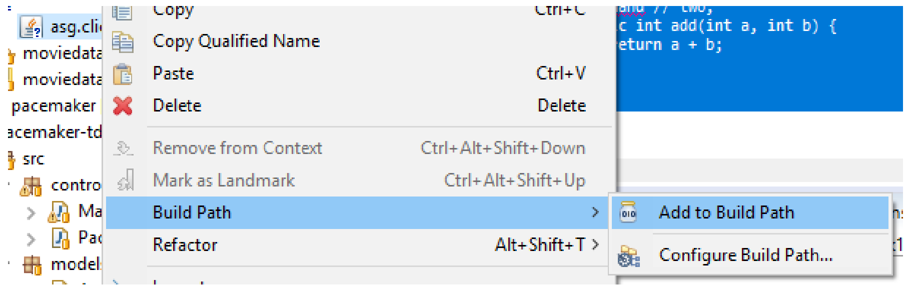
- Cliche is a small Java library enabling simple creation of CLIs (CLI stands for "command-line user interface")
- You need to use @annotations
- You need to include the cliché jar in your project
- You don't have to organise command loop or write complicated parsers/converters for primitive types



abbrev	name	params
!rs	!run-script	(filename)
!el	!enable-logging	(fileName)
!dl	!disable-logging	()
!gle	!get-last-exception	()
!sdt	!set-display-time	(do-display-time)
?la	?list-all	()
?l	?list	()
?ghh	?generate-HTML-help	(file-name, include-prefixed)
?l	?list	(startsWith)
?h	?help	()
?h	?help	(command-name)

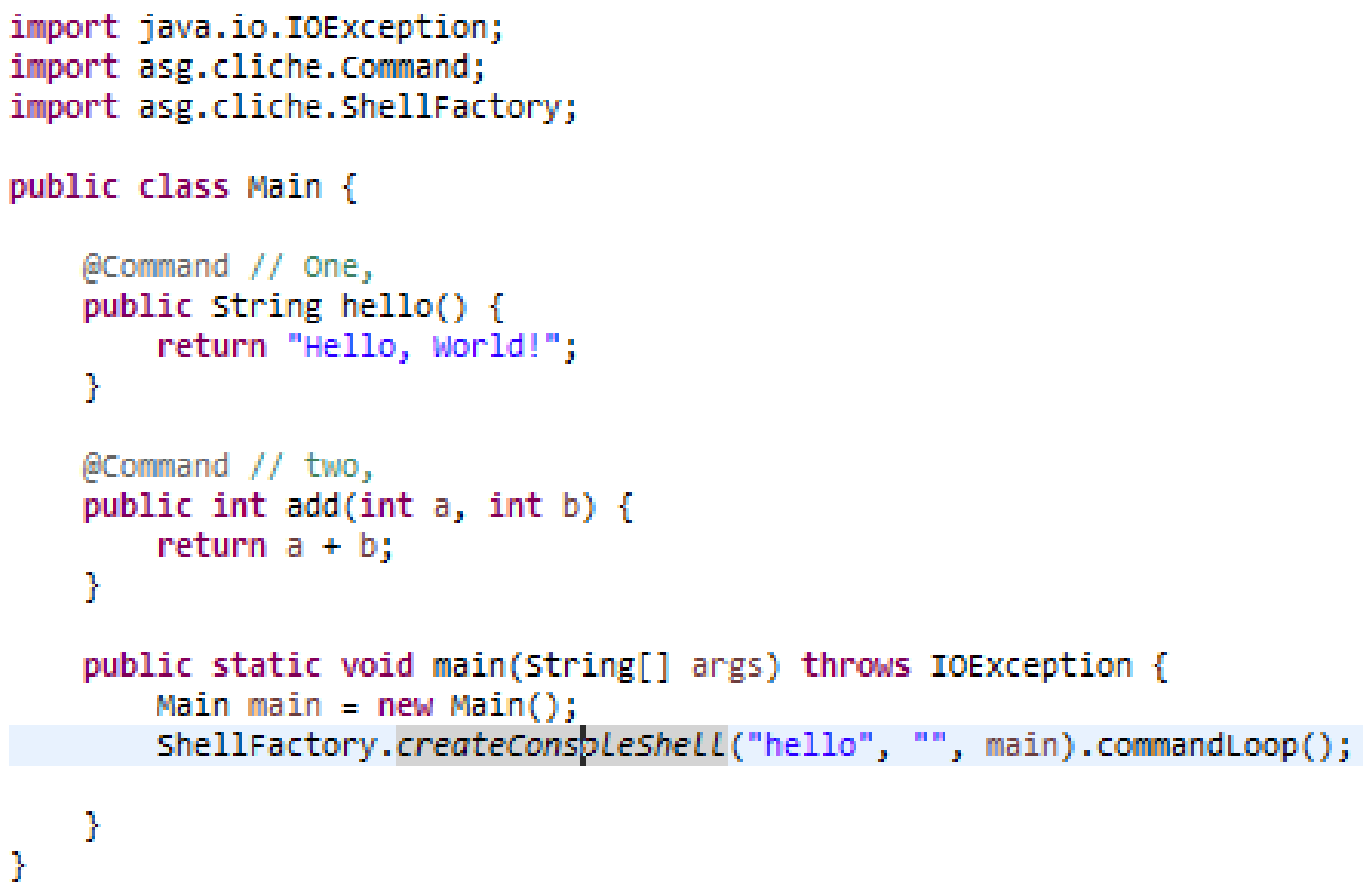
Steps to include Cliche

- Get the Jar from here
- Add the jar to the class path of your project:
  - Create a folder called lib in your project (if there isn’t one already)
  - Copy the jar to the lib folder
  - Right-click on the jar and select “Build path”->“add to Buildpath”



```
c int add(int a, int b) {
eturn a + b;
```

@annotate



```java
import java.io.IOException;
import asg.cliche.Command;
import asg.cliche.ShellFactory;

public class Main {

    @Command // One,
    public String hello() {
        return "Hello, World!";
    }

    @Command // two,
    public int add(int a, int b) {
        return a + b;
    }

    public static void main(String[] args) throws IOException {
        Main main = new Main();
        ShellFactory.createConsoleShell("hello", "", main).commandLoop();

    }
}
```

Simplest usage

- Mark methods with @Command and run the commandLoop()
- ShellFactory used to create the console shell:

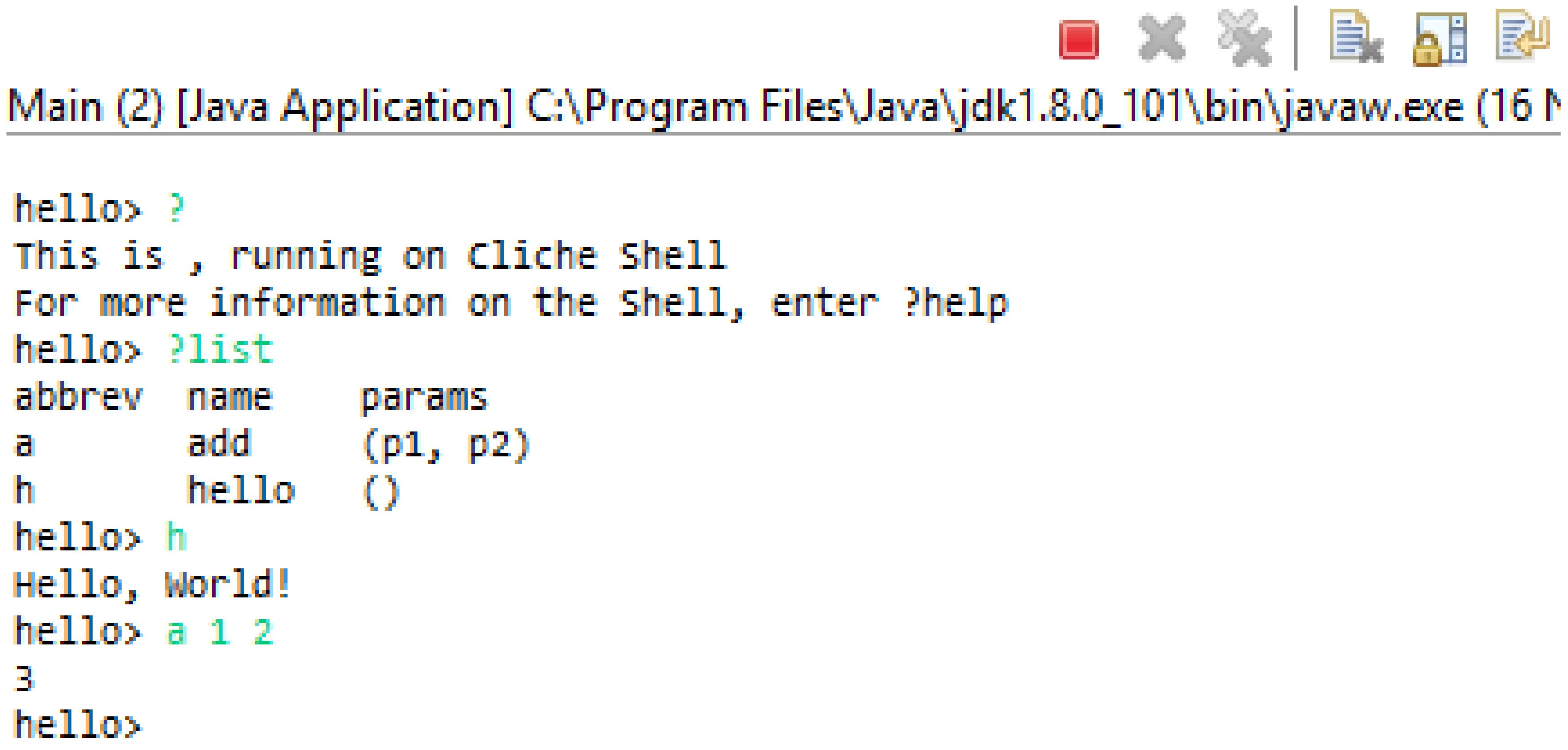
```
ShellFactory.createConsoleShell("hello", "", main).commandLoop();
```

text displayed in the command prompt

text displayed upon startup and every time user enters `?` command.

objects whose annotated methods are to be turned into commands

Running Console



Main (2) [Java Application] C:\Program Files\Java\jdk1.8.0_101\bin\javaw.exe (16 M

```
hello> ?
This is , running on Cliche Shell
For more information on the Shell, enter ?help
hello> ?list
abbrev  name    params
a       add     (p1, p2)
h       hello   ()
hello> h
Hello, World!
hello> a 1 2
3
hello>
```

@Command

- To make it well-documented command you transform these comments into annotation parameters:

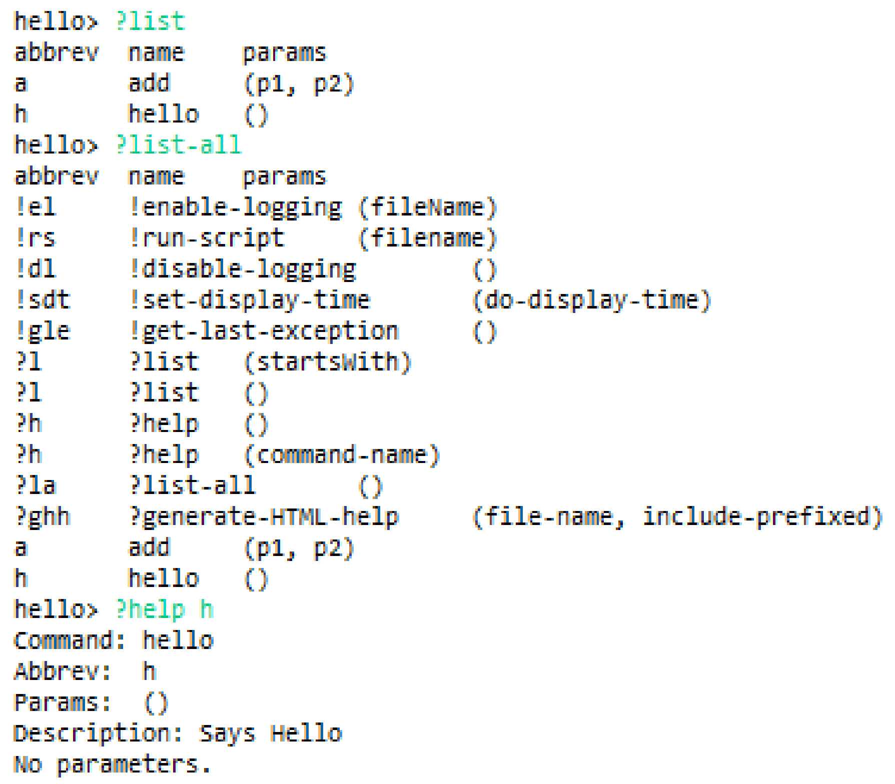
  @Command(description=“Says Hello")

- To change the auto generated name or abbreviation of the command

@Command(name="hello", abbreviation="h1")

HELP!

- There are three help commands: ?list, ?list-all and ?help.



```
hello> ?list
abbrev  name    params
a       add     (p1, p2)
h       hello   ()
hello> ?list-all
abbrev  name    params
!el     !enable-logging (fileName)
!rs     !run-script     (filename)
!dl     !disable-logging        ()
!sdt    !set-display-time       (do-display-time)
!gle    !get-last-exception     ()
?l      ?list   (startsWith)
?l      ?list   ()
?h      ?help   ()
?h      ?help   (command-name)
?la     ?list-all       ()
?ghh    ?generate-HTML-help     (file-name, include-prefixed)
a       add     (p1, p2)
h       hello   ()
hello> ?help h
Command: hello
Abbrev:  h
Params:  ()
Description: Says Hello
No parameters.
```