

# Practical Java

Kristof Werling

April 2022

# Organizational Info

# Each day of the lecture

- 2 Parts
  - Part 1:
    - Learn about new Java concepts and functionality
    - Easy and short exercises to enhance the understanding of the material
  - Part 2:
    - Use the new concepts and functionality of Part 1 to enhance and extend the Converter project

# Part 1 - more details

- There will be exercises
- Each exercise comes with a solution
- Each exercise will be discussed with the whole group and problems / issues will be addressed
- The solution provided also will be discussed

# Depth of the information provided

- For the most part the information provided is sufficient to work out the solution to the exercises
- For most of the concepts and functionality shown there is a vast body of knowledge we cannot explore in any kind of practical manner.

## Project for this lecture

- Converter: Markdown to Latex and later to Html
- Simple solution.
- Each lecture works on one aspect of the solution (like: GUI, DB, ...)

# Markdown Tags

Headings	Text Formatting	Rendered Output
# Heading level 1	<b>**This is bold text**</b>	Link to [Google](https://www.google.com/)
## Heading level 2	<b>__This is bold text__</b>	- Unordered List Item 1
### Heading level 3	<i>*This text is italicized*</i>	- Unordered List Item 2
#### Heading level 4	<del>~~Strikethrough text~~</del>	- Unordered List Item 3
##### Heading level 5	<b><i>***Bold and italics text***</i></b>	1. Ordered List Item 1
##### Heading level 6	<b><i>**Bold and *nesting italics* text**</i></b>	1. Ordered List Item 2

Source: <https://www.markdownguide.org/basic-syntax>

Or

<https://docs.github.com/en/get-started/writing-on-github/getting-started-with-writing-and-formatting-on-github/basic-writing-and-formatting-syntax>

All the gory details (specs): <https://github.github.com/gfm/>

# Latex

## Boiler Plate for a Latex Document

```
\documentclass[12pt, a4paper] {article}
```

```
\begin{document}
```

Here goes the document.

```
\end{document}
```

Source: [https://www.overleaf.com/learn/latex/Learn\\_LaTeX\\_in\\_30\\_minutes](https://www.overleaf.com/learn/latex/Learn_LaTeX_in_30_minutes)



# Latex

Bold	This is <code>\textbf{bold text}</code>
Italic	This is <code>\textit{text in italic}</code>
Strikethrough	
Unordered List	<code>\begin{itemize}</code>
	<code>\item Item 1</code>
	<code>\item ...</code>
Ordered List	<code>\end{itemize}</code>
	<code>\begin{enumerate}</code>
	<code>\item Item 1</code>
	<code>\item ...</code>
Link	Use the <code>hyperref</code> package

Source: [https://www.overleaf.com/learn/latex/Learn\\_LaTeX\\_in\\_30\\_minutes](https://www.overleaf.com/learn/latex/Learn_LaTeX_in_30_minutes)

# Latex

Heading 1	<code>\section{section}</code>
Heading 2	<code>\subsection{subsection}</code>
Heading 3	<code>\subsubsection{subsubsection}</code>
Heading 4	<code>\paragraph{paragraph}</code>
Heading 5	<code>\subparagraph{subparagraph}</code>

Source: [https://www.overleaf.com/learn/latex/Learn\\_LaTeX\\_in\\_30\\_minutes](https://www.overleaf.com/learn/latex/Learn_LaTeX_in_30_minutes)

## What we need to get started

- IntelliJ Community Edition ([Download here](#))
- MiKTeX (or any other Tex that can process Latex) ([Download here](#))
- OpenJDK Java 18 ([Download here](#))
- Markdown Viewer ([For example: Windows Markdown Viewer](#))
- Git for Windows ([Download here](#))

## Github.Com Account



Product ▾ Team Enterprise Explore ▾ Marketplace Pricing ▾

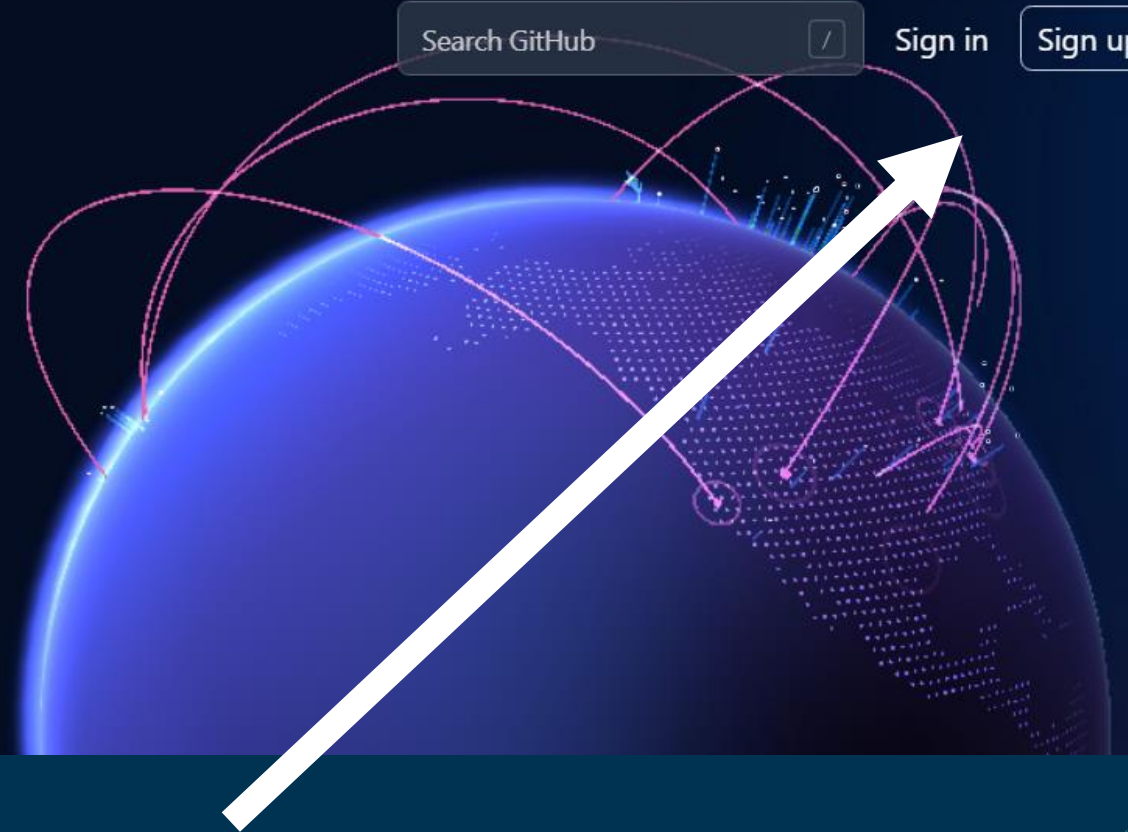
Search GitHub



Sign in

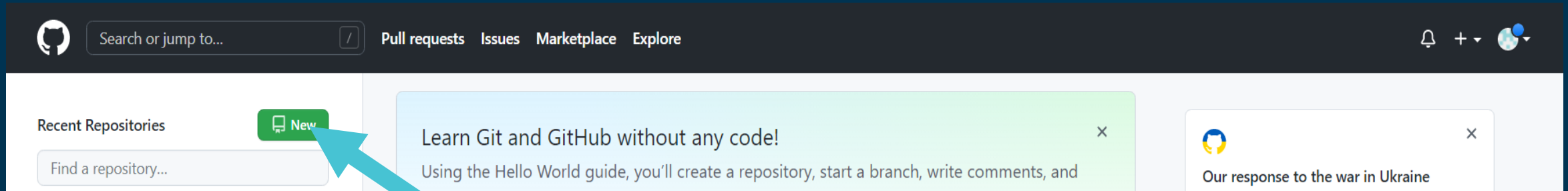
Sign up

# Where the world builds software



Sign in or Sign up

## Create new Repository



**New Repository**

# Create new Repository 1 of 2

## Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere?  
[Import a repository.](#)

Owner \*

kwerling-MM

Repository name \*

myName

Great repository names are short and memorable. Need inspiration? How about [fuzzy-committing machine?](#)

Description (optional)

This is the comment to my new Repository

☐



Public

Anyone on the internet can see this repository and commit to it.

☒



Private

You choose who can see and commit to this repository.

Repository name

Comment, if wished

Private or Public access

# Create new Repository 2 of 2


Initialize this repository with:  
Skip this step if you're importing an existing repository.


☐ Add a README file  
This is where you can write a long description for your project. [Learn more.](#)

☒ Add .gitignore  
Choose which files not to track from a list of templates. [Learn more.](#)  

.gitignore template: Java ▼

☐ Choose a license  
A license tells others what they can and can't do with your code. [Learn more.](#)

This will set  main as the default branch. Change the default name in your [settings](#).

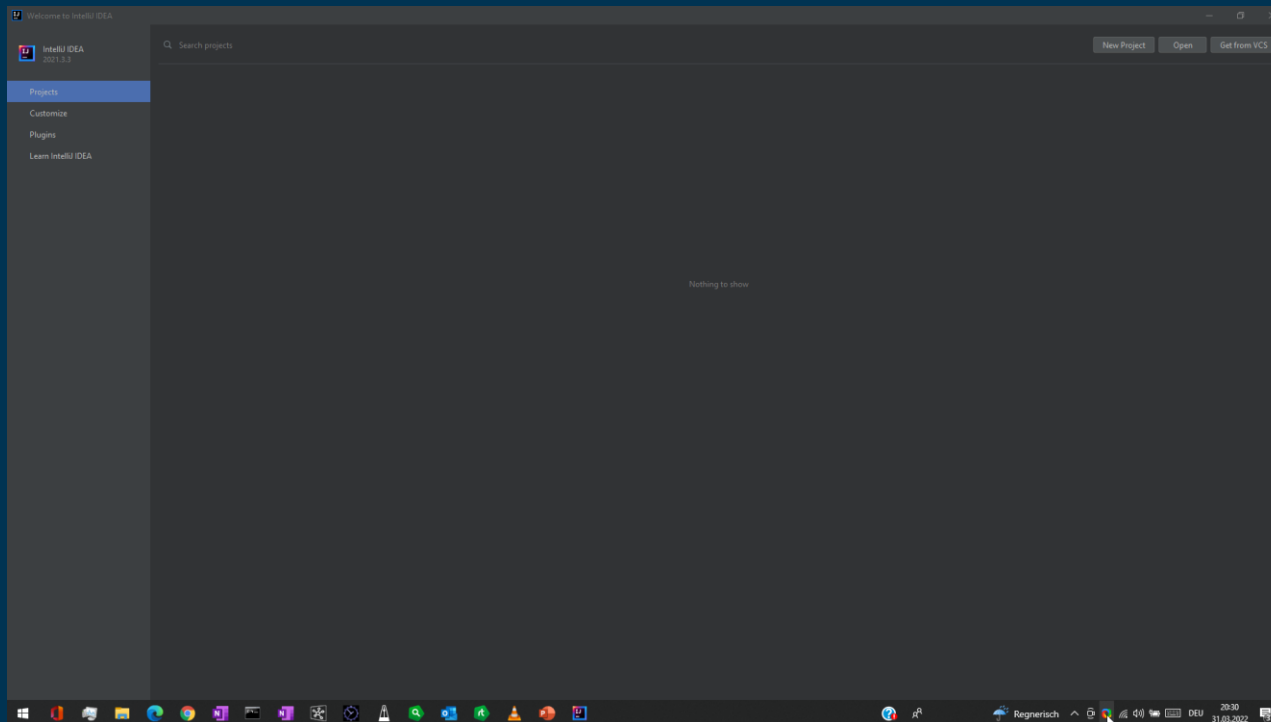
 You are creating a private repository in your personal account.

Create repository

Add .gitignore for JAVA

Create it

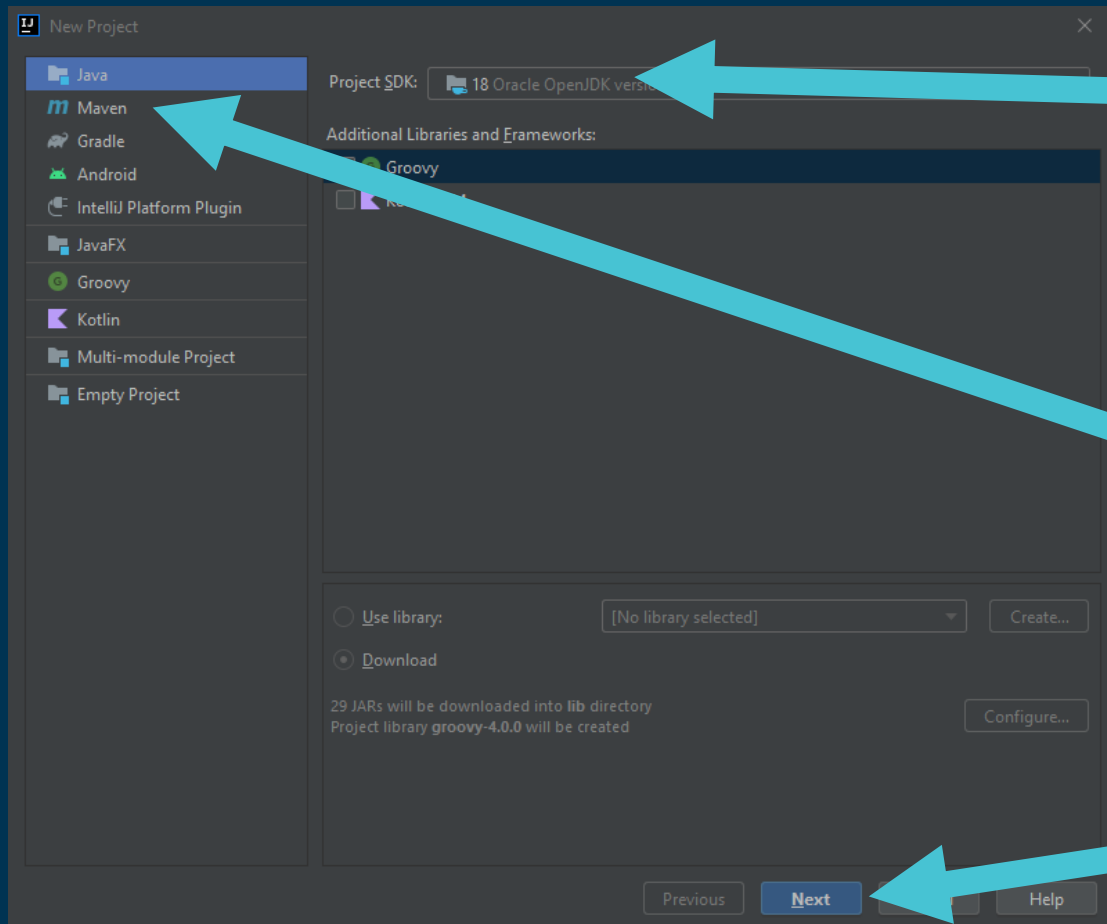
## Creating a new project 1 of 5



**New Project**



## Creating a new project 2 of 5

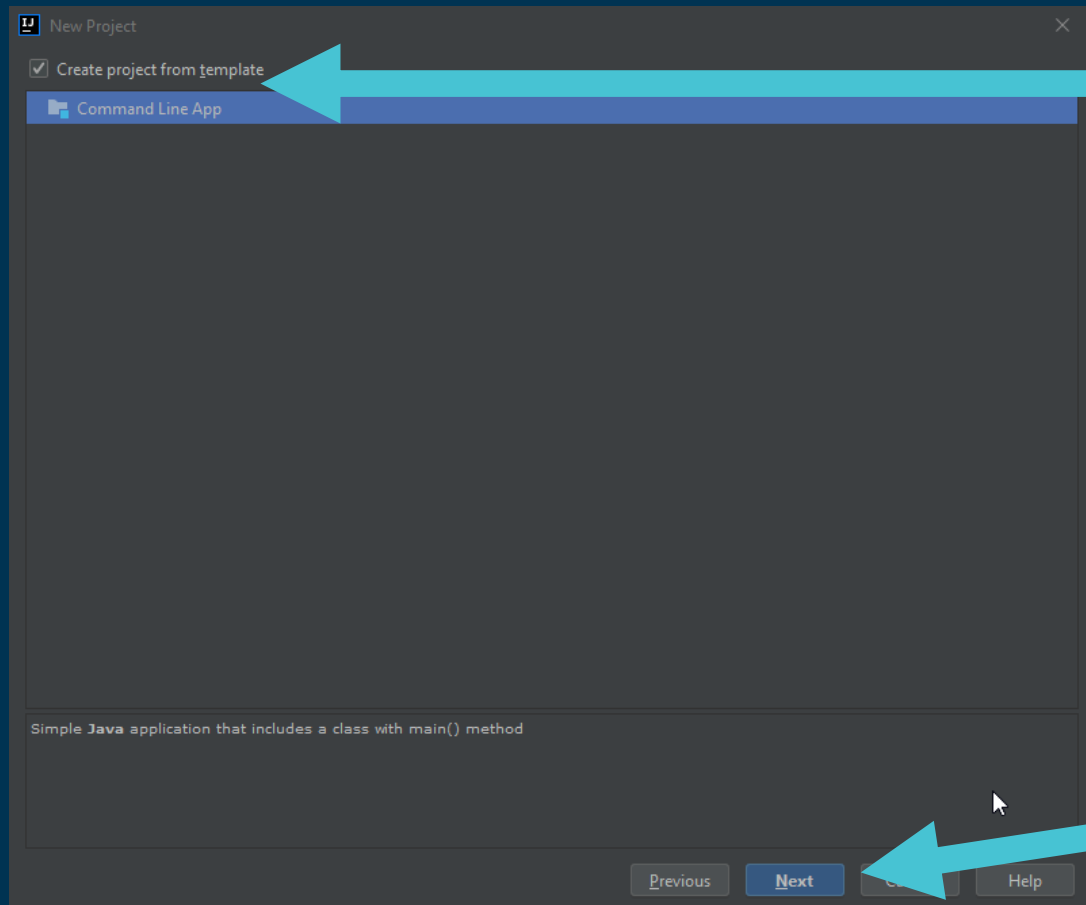


**Make sure your  
JDK is listed  
here**

**Java**

**Press Next**

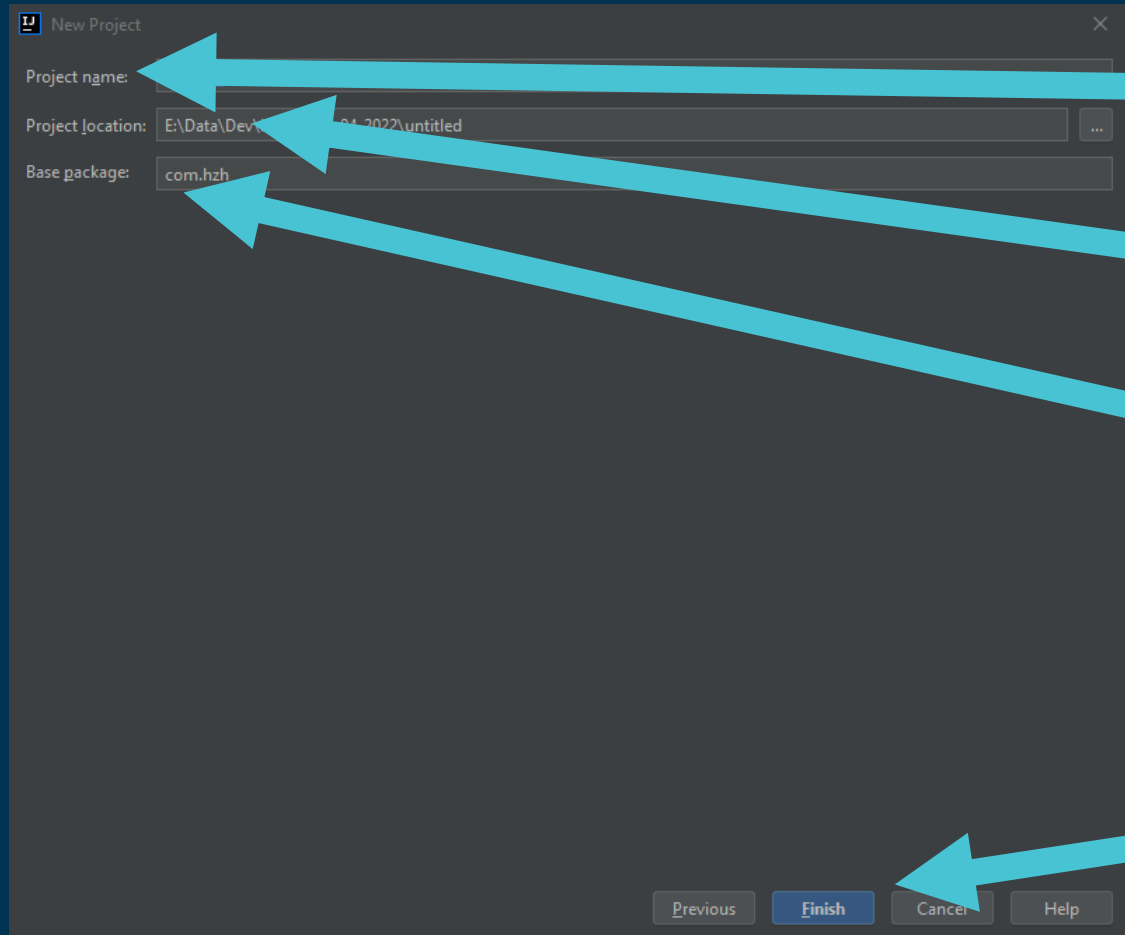
## Creating a new project 3 of 5



**Choose the  
Commend Line  
Template to get  
started**

**Press Next**

## Creating a new project 4 of 5



The screenshot shows the 'New Project' dialog box with the following fields and buttons:

- Project name:** An empty text field.
- Project location:** A text field containing 'E:\Data\Dev\... 04-2022\untitled' with a browse button (three dots) to its right.
- Base package:** A text field containing 'com.hzh'.
- Buttons:** 'Previous', 'Finish' (highlighted in blue), 'Cancel', and 'Help'.

Four red arrows point from text labels on the right to specific parts of the dialog:

- From 'Give it a name' to the 'Project name' field.
- From 'Local directory' to the 'Project location' field.
- From 'Name space' to the 'Base package' field.
- From 'Press Next' to the 'Finish' button.

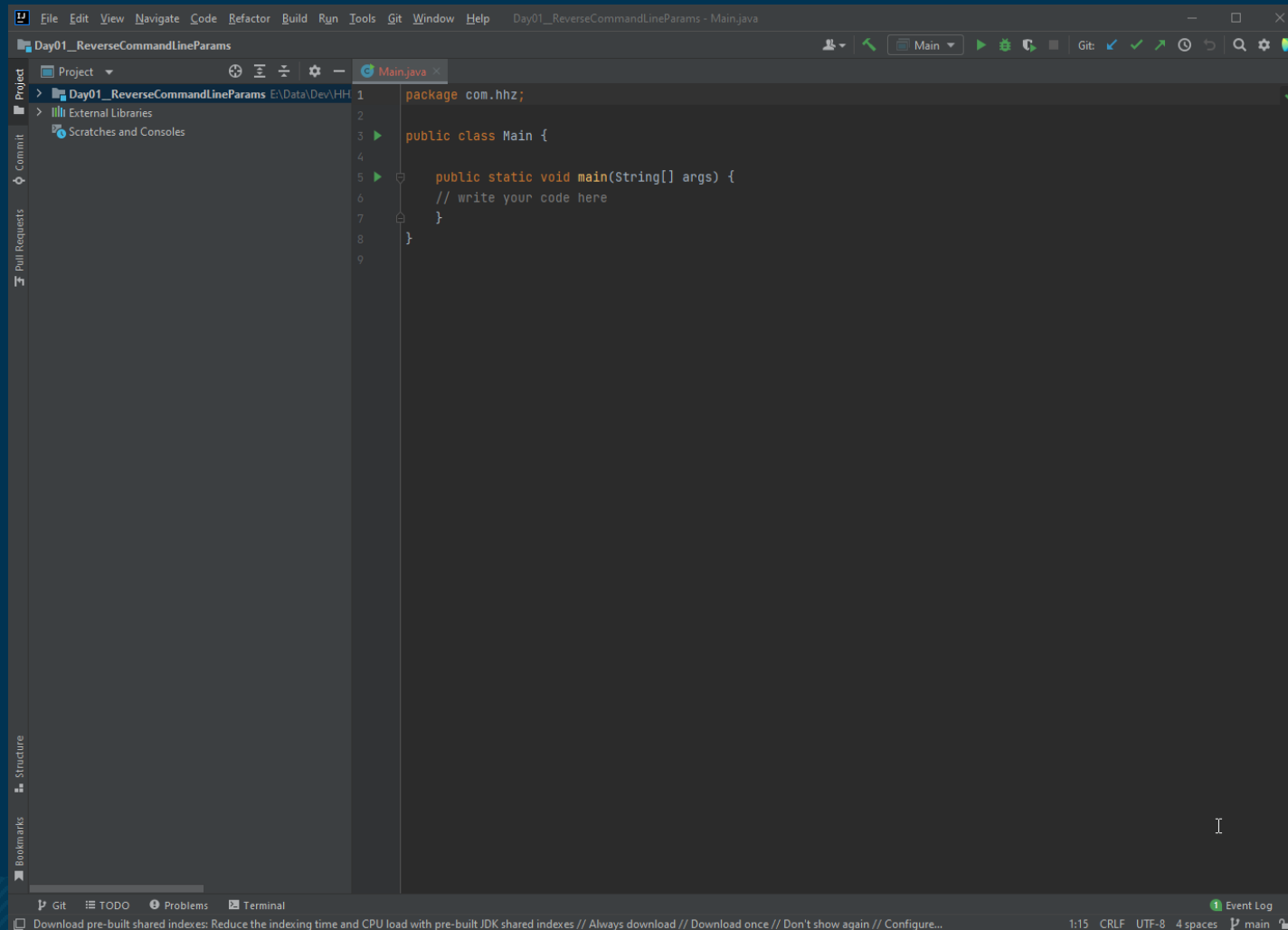
**Give it a name**

**Local directory**

**Name space**

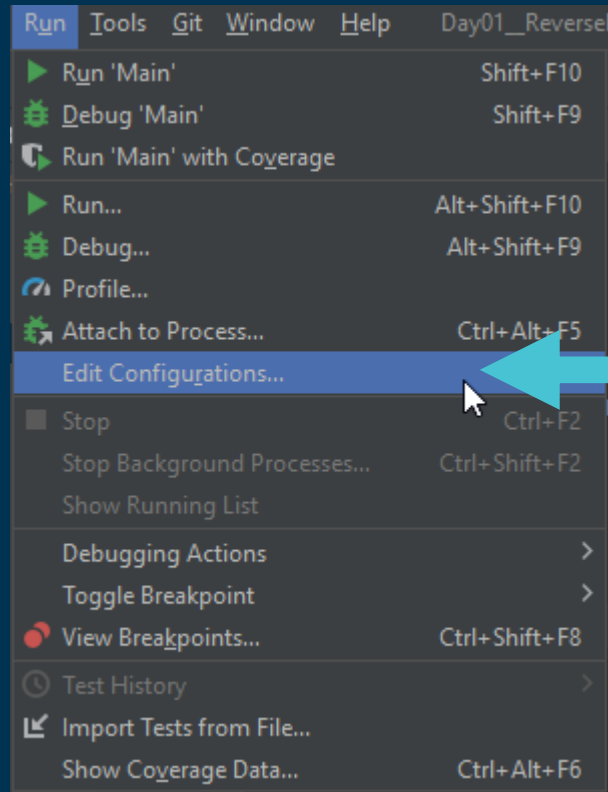
**Press Next**

## Creating a new project 5 of 5



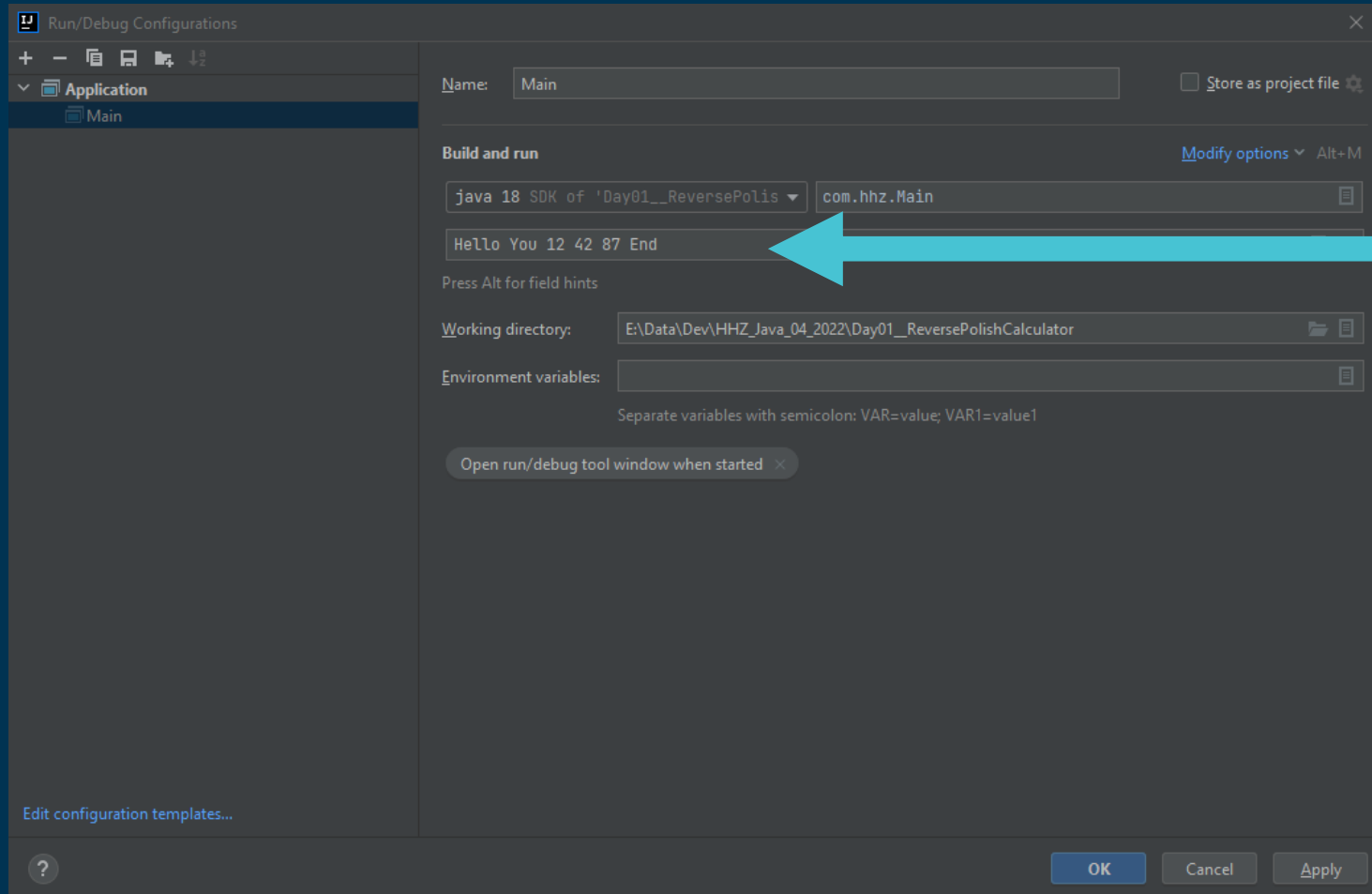
## Resulting project

# Run with Command Line Params 1 of 2



**Adjust the configuration**

# Run with Command Line Params 1 of 2



Here go the command  
Line params

# Table of Content

- Project used in the lecture
- Day 1: Playing with Java / Deepen the knowledge
- Day 2: User Interfaces
- Day 3: Networking – From Socket to Message Bus
- Day 4: Working with Databases ( SQL and No-SQL )
- Day 5: Wrap-Up and Overflow

# Command line parameters

- The main function takes the command line parameters in an String array
- Each parameter is passed on as a String (String array after all)
- In case of no command line parameters the String array is empty



## Exercise 01

- Write a program, which prints out the command line parameters in reverse order

# Exercise 01 - Solution

```
1 package com.hhz;  
2  
3 public class Main {  
4  
5     public static void main(String[] args) {  
6         for( int i = args.length; i>0; i--) {  
7             System.out.println("Param #" + i + ": " + args[i-1]);  
8         }  
9     }  
10 }  
11
```

# Some methods of the String class

(some) String class method	Functionality
<u><a href="#">String toLowerCase()</a></u>	It returns a string in lowercase.
<u><a href="#">String toUpperCase()</a></u>	It returns a string in uppercase.
<u><a href="#">String trim()</a></u>	It removes beginning and ending spaces of this string.
<u><a href="#">int indexOf(String substring)</a></u>	It returns the specified substring index.
<u><a href="#">String[] split(String regex)</a></u>	It returns a split string matching regex.
<u><a href="#">boolean contains(CharSequence s)</a></u>	It returns true or false after matching the sequence of char value.
<u><a href="#">int length()</a></u>	It returns string length. Compare to Array.length !!
<u><a href="#">String substring(int beginIndex, int endIndex)</a></u>	It returns substring for given begin index and end index.

## Exercise 02 - Playing with String comparison

- Take the code on this slide
- Run it
- Explain the results



Main.java

## Exercise 03 - Playing with String concatenation

- Take the code on this slide
- Run it
- Explain the results



Main.java