

## Exercises about Windows Forms and WPF

- Solve them in Visual Studio.

### *Exercise 16.01: Replicate some Forms*

- Create a Windows Forms or a WPF application.

There are a lot of examples where a form is used. Most examples do have a Windows Forms version and a WPF version.



*Recreate the examples from scratch.*

*This can be very time consuming, so do these exercises for the functionalities that you want to exercise.*

*The layout hasn't to be pixel perfect, but the naming of the controls / elements and the functionality must be correct.*

*Don't do exercises where you don't learn anything, except if you want to get routine in how to do things.*

### The list is very long

- 00001-d SimpleProject.zip.
- 00002-a FirstForm.zip.
- 00002-b FirstSolution.zip.
- 00002-c Program.zip.
- 00006e ConstantTester.zip.
- 00010-r Factorial.zip.
- 00010-s Overload.zip.
- ...
- Everything in 00060 (only WPF) (some are very advanced)

Notes

---

---

---

---

---

---

---

---

---

---

COPY PASTE)

## Exercise 16.02: Save and Open some Text File

- Create a Windows Forms or a WPF application.
- Create a form with 4 components.
  - Every component must have a good name.
  - 1 textbox (To type text in).
  - 2 buttons (To click on).
  - 1 label (To show the result).
- In the textbox you can type text, multiple lines even with hard returns (enters) in it.
- One button, takes that typed text and saves it to a file (somewhere on your hard disk)
- The other button, reads the content of that saved file and shows it on the label.



*You are responsible for clean code, if you have code and for the classes, if you have classes.*

*The general rule is, that every change can be done at one place.*

*Make sure that your application is monkey proof. I will try to break your application.*

*For example:*

- *I will hit the button that reads the text, even if the file is not there.*

### Variant 1

- Add a textbox, where you can enter the full path of the text file.

Notes

.....

.....

.....

.....

.....

.....

.....

COPY PASTE)

## *Exercise 16.03: Track the Windows*

- The starting point is 00060-g WindowTracker.zip.
  - A window with 2 buttons.
  - Clicking on one button, creates a new window.
  - Clicking the other button, changes the content of all started windows.

### **Variant 1**

- Change a WPF application.
- The title (caption) of created windows must contain a timestamp when the window was created.
- In the content of the created window, you also show how many times, the time has been refreshed on that window.
  - Attention point: This can be different for the windows.

### **This is what I will do to test**

- I start the application.
- I create several windows.
- I refresh the time.
- I create several windows.
- I will stop some windows.
  - Will this work without finding a solution for variant 2?
  - Why will it (not) work?
- I refresh the time.

### **Variant 2**

- When you close a created window, it must be removed from the list of started windows.
  - Tip: Working with a list is not the best option here
    - There are structures / data types that are better suited for this kind of functionality.

Notes

---

---

---

---

---

---

---

---

COPY PASTE)

### *Exercise 16.04: What is Threading anyway?*

- The starting point is 00060-k Multithreading.zip.
  - An application that has 2 threads and you can cancel the action of the thread with a Cancel button.
- Create a new example, but only with the minimum amount of code you need to have a working example.
- In your future programming, you will encounter this pattern a lot.
- The goal is to have a minimum starting point to build the other projects on. A kind of template.

#### **Variant 1**

- The work can be a counter, till 500.000.000.
- The moment you click “Cancel” the value of the counter can be displayed on the form.
- The progress bar can be progressing very slowly, every time you count passes 5.000.000, a percent is added to the progress bar.
  - This result resembles a lot to 00060-i BackgroundWorker, but has a progress bar.
  - This example uses the WorkerReportsProgress property, while the Dispatcher is used in 00060-i BackgroundWorker.



*Study the different way of working.*

- *Using the Dispatcher.*
- *Using the WorkerReportsProgress Property.*

Notes

---

---

---

---

---

---

---

---

COPY PASTE)

## Variant 2

- Experiment with this pattern.
  - By trying to add a second BackgroundWorker element.
- Check which task is finished first.
- And so on ...



*This is a basic pattern used in games.*

*You are a game character (Object) and you are shot (Event)*

- *You are bleeding (in the background)*
- *You are trying to heal yourself (in the background)*

*What happens first? You are bleeding to death or you are healed.*

## Variant 3

- Rewrite 00060-i BackgroundWorker, but use the property WorkerReportsProgress instead of the Dispatcher.

Notes

---

---

---

---

---

---

---

---

COPY PASTE)

*Exercise 16.05: An Easy Calculator, really?*

We create the basic functionality of an easy calculator.

How it looks is not important, make that you can calculate with it, according to the asked specifications.

**Basic functionality (simplified)**

- You click on numbers to type a number (decimals are possible).
- Then you click “division”, “multiplication”, “minus” or “add”.
- You click on numbers to type another number (decimals are possible).
- Then you hit “=”.
- You see the result.
  - This result disappears the moment you start clicking on another button.



*All buttons should trigger the same event.*

*Meaning. There is only one method that is starting point for all clicks. Submethods are possible, but only one method is handling the click of the buttons.*

*Think: Bubbling up.*

**Variant 1**

- When the calculator has its basic functionality, you invent other functionality.
- This can be a very lengthy style exercise.
- Mention what your goal is in the comment of your code, then execute.

Notes

---

---

---

---

---

---

---

---

COPY PASTE)

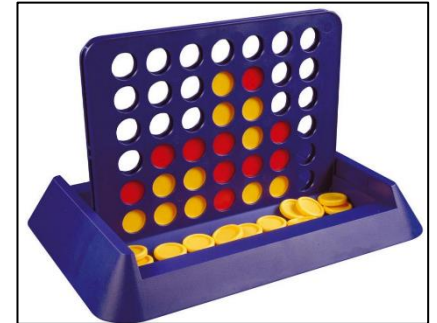
*Exercise 16.06: Playing Four in a Row*

*This exercise will take time. So manage your time, do the stuff where you learn something.*

*First on paper, comment what you want to do first, before you start implementing.*

Create the game: “Four in a row”.

- There is a grid 7 wide and 6 high.
  - Find out for yourself what are the best controls do handles this.
- There are 2 colours.
- Randomly a colour begins.
- When you click on a column, that colour falls down till the lowest empty position.
- Then you switch colours.
- The game is finished when there are 4 stones of the same colour in the same row, column or diagonal.



*Solutions can be found on the internet, but they are more complicated than asked in this exercise.*

*For example: [https://github.com/rinaok/4\\_in\\_a\\_row](https://github.com/rinaok/4_in_a_row).*

*Make sure you understand what happens in the code, when you Copy Paste code. Downloading this code, has already a problem at the start 😊.*

*Try to find a solution and implementation by yourself, but looking at examples will help you on the thinking process.*

Notes

---

---

---

---

---

---

---

---

COPY PASTE)

*Exercise 16.07: Reset Playing Four in a Row*

Building further on: “Four in a row”.

- The previous exercise must not be finished to tackle this problem.
- Create a custom command in your application that can be triggered in 3 ways
  - Reset the game with a menu.
  - Reset the game with a button.
  - Reset the game with a key combination.

**What is resetting the game?**

- The game is reset to its starting position.
- Empty board.
- A chosen default starting colour.



*All ingredients (tools) to do this can be found in example.*

*00060-n ApplicationAndCustomCommands.zip*

*The only thing that is different is the action bound to the menu, button and key combination.*

Notes

---

---

---

---

---

---

---

---

---

---

COPY PASTE)

## Exercise 16.08: Changing the Image

Starting point is 00062-g WPFIImage.zip.

In this WPF screen, you have 3 images, every image contains another picture, and by clicking it, the visibility of the images is changed.

So clicking on a picture, makes another picture visible.

### Your task: Making a variant

Let there be only one picture, but clicking it changes the source of the picture.



*Every resource needed can be found in the resources directory.*

*There is also in the Properties directory a class where you have properties defined towards the 3 resources.*

*Try to find a working solution on internet and implement it in this exercise.*

*Test your stuff, before you send it to me.*

Notes

.....

.....

.....

.....

.....

.....

.....

COPY PASTE)

## *Exercise 16.09: Slide the Colour away*

Starting point is 00062-i WPFSlider.zip.

In this WPF screen, you have 4 sliders. With the 4 sliders, you can change the colour of a rectangle.

### **Your task: Making a variant**

Use for the 4 sliders, the same event.



*At the moment every slider has a different event.*

*There are several different solutions to implement this.*

*Test your stuff, before you send it to me.*

Notes

---

---

---

---

---

---

---

---

COPY PASTE)