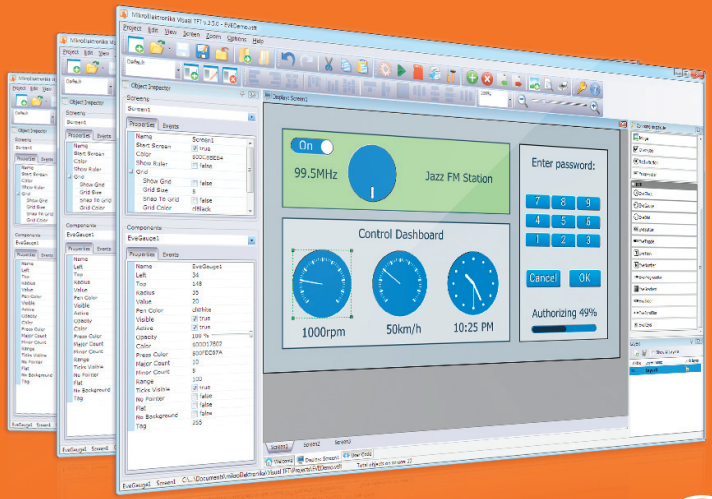
VISUAL TFT

GUI Design Made Easy



Let's go together through several easy steps and build a simple GUI with two buttons and two screens using the powerful Visual TFT software.

MikroElektronika
DEVELOPMENT TOOLS | COMPILERS | BOOKS

TFT

TO OUR VALUED CUSTOMERS

I want to express my thanks to you for being interested in our products and for having confidence in MikroElektronika.

The primary aim of our company is to design and produce high quality electronic products and to constantly improve the performance thereof in order to better suit your needs.

Nebojsa Matic
General Manager

Table of Contents

1. Introduction to Visual TFT software

Visual TFT software is a standalone application used for rapid development of graphical user interfaces for TFT displays. Software generates code compatible with mikroElektronika compilers: mikroC, mikroBasic and mikroPascal, for all supported MCU architectures: PIC, dsPIC30/33, PIC24, PIC32, ARM and AVR.

When first started, the window features following sections:

01 Main Toolbar
02 Object Inspector
03 Welcome Screen Buttons
04 Getting Started Links
05 Components Palette
06 Layers Window

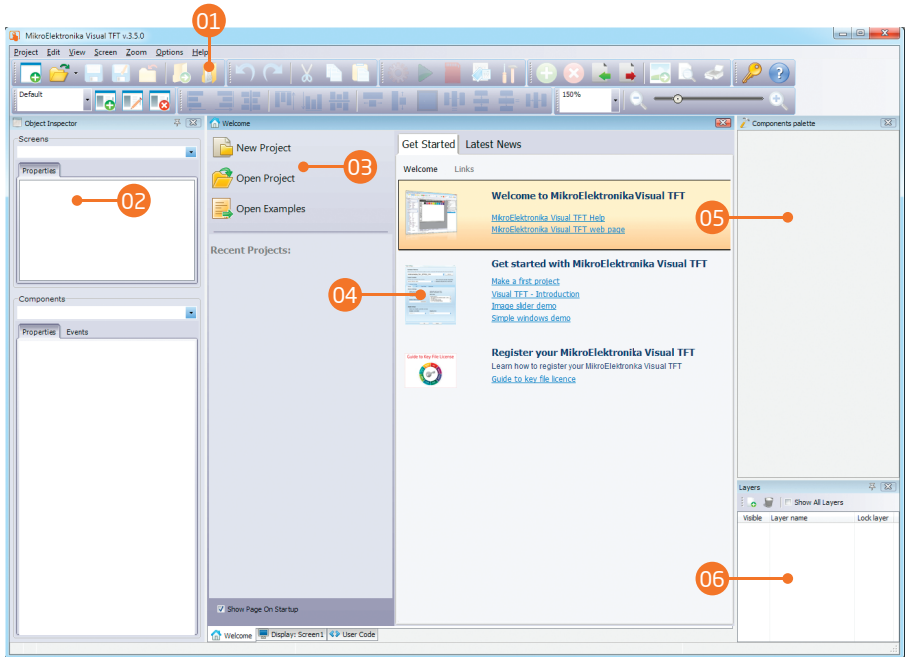


Figure 1-1: Visual TFT software displaying welcome screen when first started

2. What do we need?

Figure 2-1: mikroC PRO for PIC is a powerful ANSI C compiler for popular Microchip PIC microcontrollers

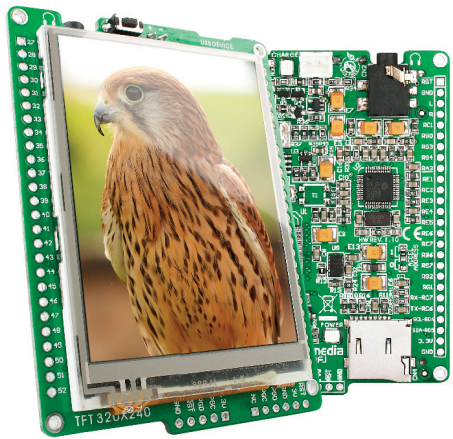
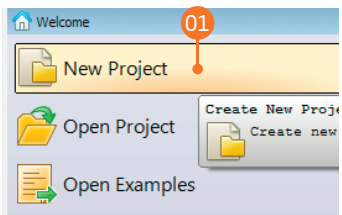
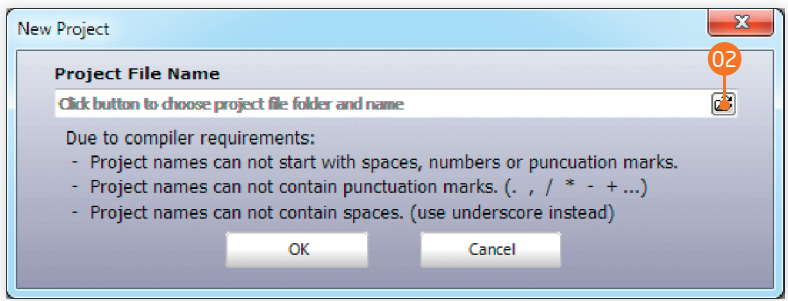


Figure 2-2: mikromedia for PIC18FJ is a PIC development board with multimedia modules

In this tutorial we will develop a simple application with two screens. Each screen will carry a button which can be used to switch to the other screen. We will be designing our graphical interface in **Visual TFT** software, and after we add user code we will use **mikroC PRO for PIC** compiler to build it. We will download the firmware to the **mikromedia for PIC18FJ** development board and test it on 320x240px TFT with Touch Panel. Let's begin!

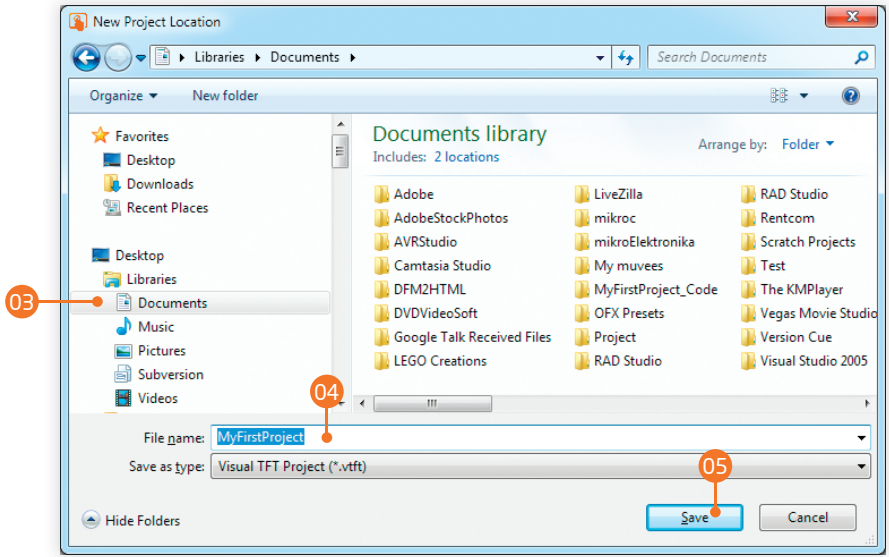
3. New Project Wizard





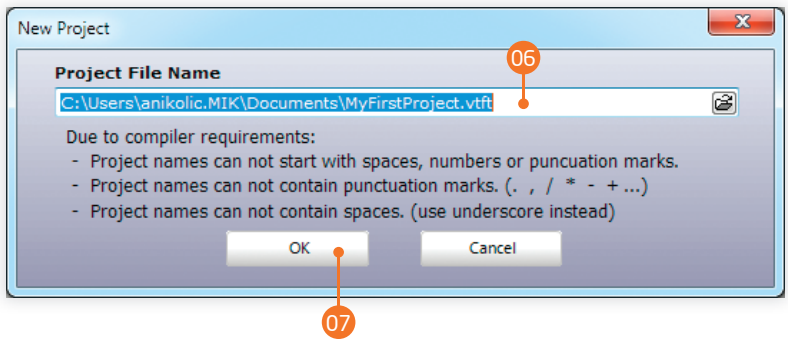
Start a new project wizard

Let's start by creating a new project. On the welcome screen click the **New Project** button 01. A new window will appear and guide you through the process of creating a new project. The first thing we need to do is to specify the new project's name and destination folder. Click the browse button 02 next to the edit field.



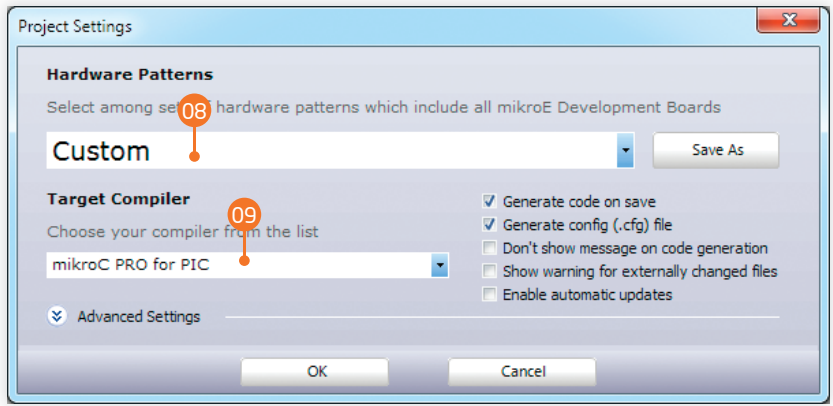
Specify project name and location

A new dialog window will appear. First select the destination folder 03 where you want to store your new project. Then specify the project's name 04. **"MyFirstProject"**, for example. Then click the **Save** button 05 to confirm.



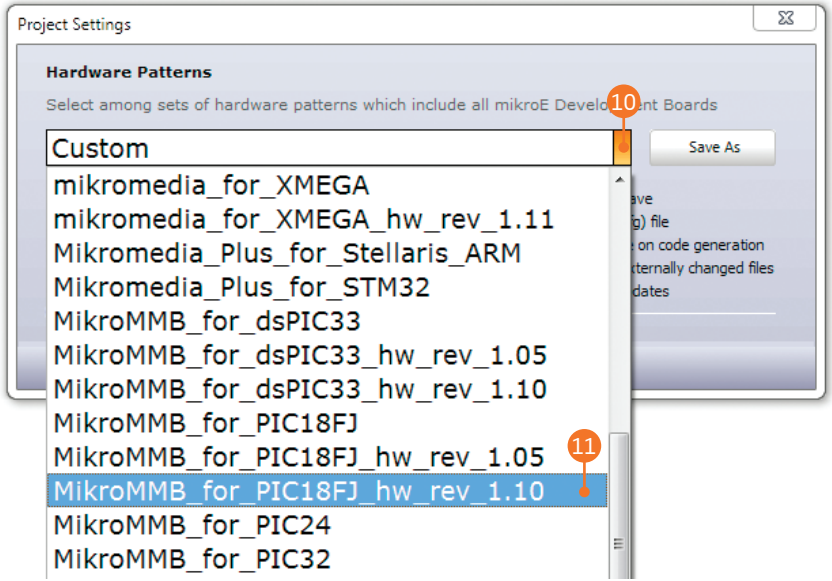
Make the final check and create a new project

Full project path will be shown in the edit field 06. If you want to change destination path or project name you can still do it. When you are done click the **OK** button 07 to create a blank new project.



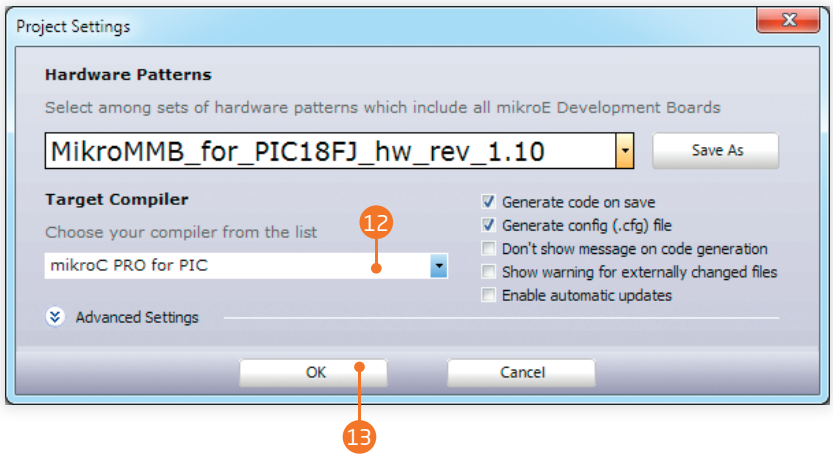
Quick project configuration using Project Settings

After the project is created the **Project Settings window** will appear. We need to specify the target hardware we will be using 08 and compiler 09 as well.



Select the target hardware

Click the button 10 of the first dropdown box and a list of hardware configuration patterns will appear. Each one carries complete settings of TFT and Touch Panel connections for the target hardware. We will select **MikroMMB_for_PIC18FJ_hw_rev_1.10** from the list 11.



Select the target compiler and confirm

Now we need to select the target compiler. We will only be able to choose among PIC compilers because our selected target hardware (mikromedia for PIC18FJ) is a PIC development board. Select **mikroC PRO for PIC** 12 and hit **OK** 13.

4. Designing the User Interface

So far, we have successfully created a blank new project for **mikromedia for PIC18FJ** development board. Graphics will be displayed on 320x240 pixel graphical display based on KS108 controller. A 4-wire resistive touch panel is placed on top of the display, thus creating a Touch Screen module. We have chosen to use **mikroC PRO for PIC** compiler and the code generated by Visual TFT will be compatible with it. Let's take a brief look at the **Visual TFT** window before we begin. Here are the main sections of this window:

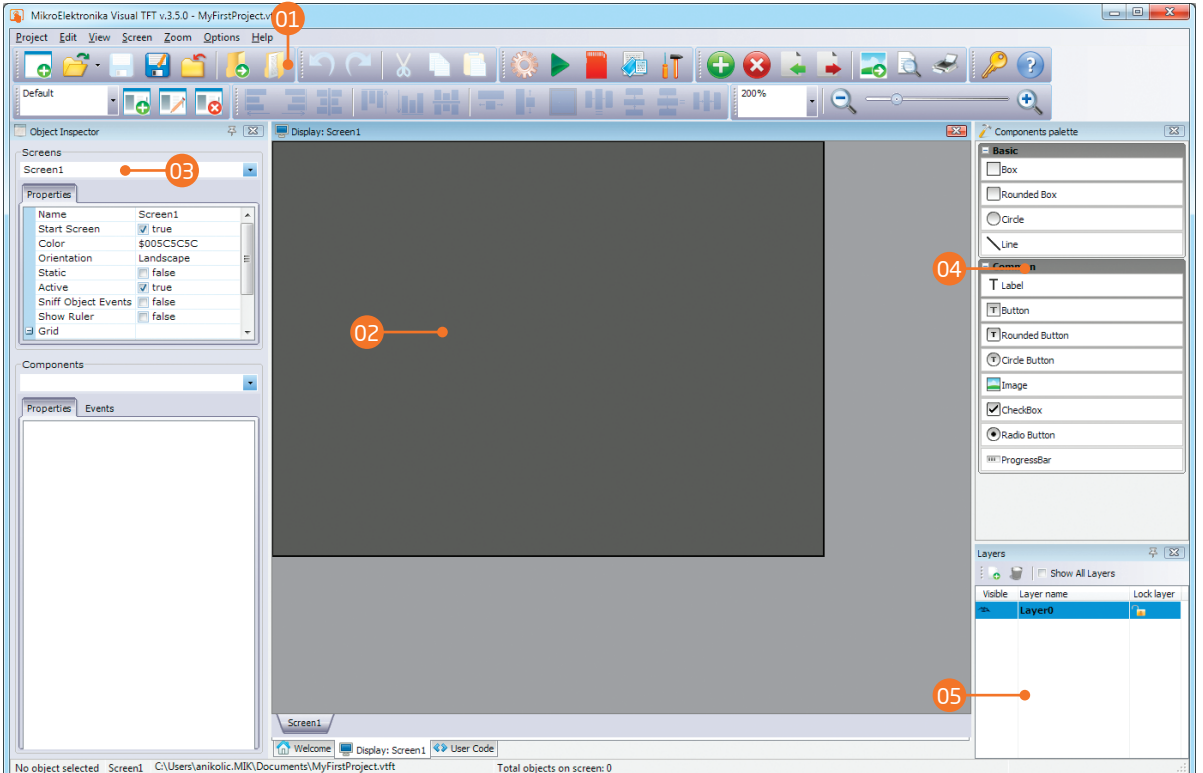
01 **Main Toolbar.** It features buttons with icons that depict each button's function. Here we can open, save or export projects, add or delete screens, generate code, start the target compiler, invoke Project Settings window and much more.

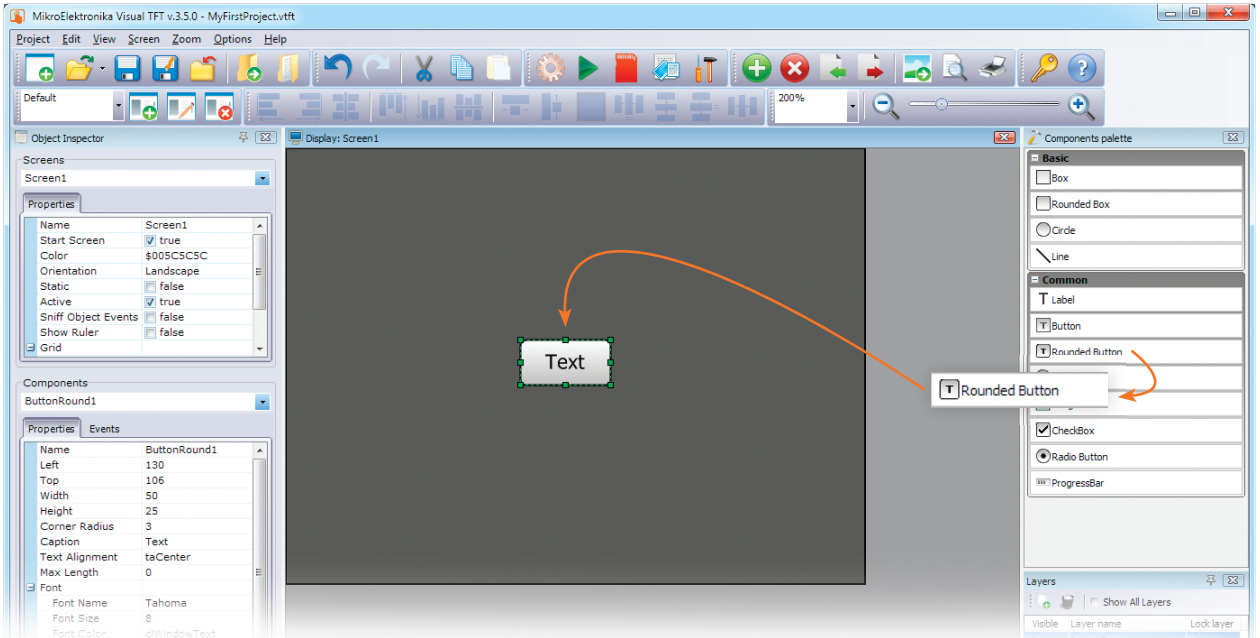
02 **Current Screen**. This is the graphical representation of the active display surface. It's the area where we will be placing components and designing graphical user interface for this project. You can add as many screens as you want. We will use just two.

03 **Object Inspector.** This window can be used to change properties of each screen and component. Change names, captions, fonts, sizes, position, add events to components (OnUp, OnDown, OnClick, OnPress) and define their behaviour.

04 **Components Pallete.** Collection of components which can be placed on screens. There are simple, basic components, such as box, circle, line, image and label, and as well as more complex components such as buttons, checkbox and progress bar.

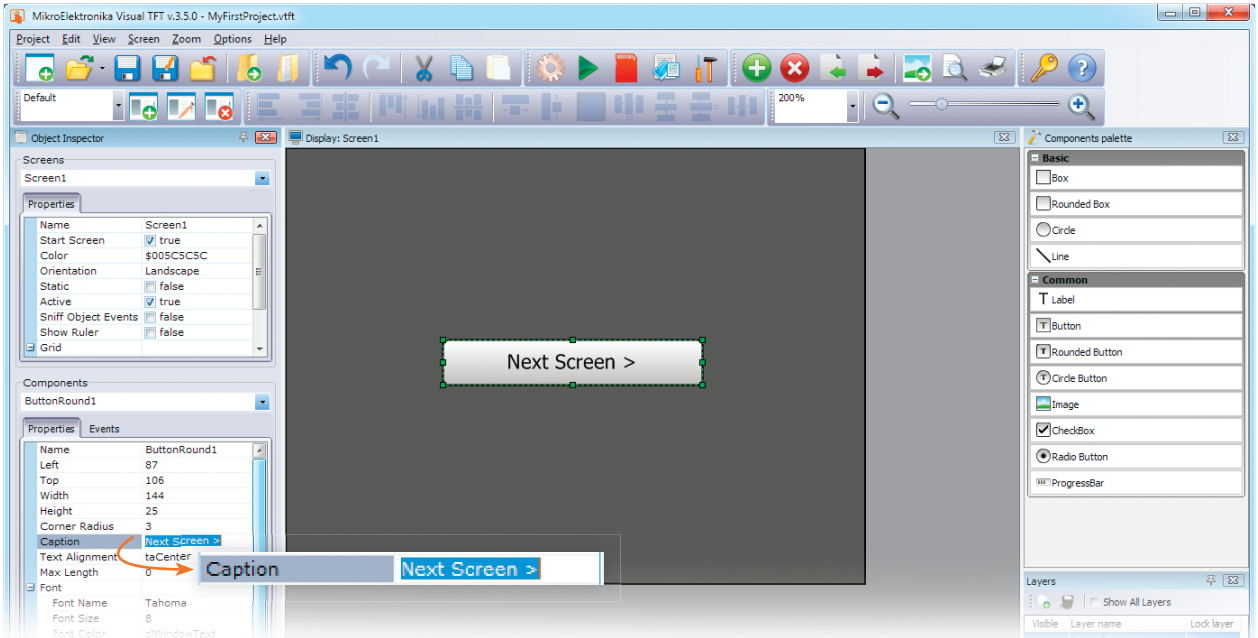
05 **Layers**. Like in any other vector graphic editor, you can group components on layers while designing, and quickly navigate through different layers using this window.


01
02
03
04
05
MikroElektronika Visual TFT v.3.5.0 - MyFirstProject.v
Project Edit View Screen Zoom Options Help
Default
200%
Object Inspector
Screens
Screen1
Properties
Name Screen1
Start Screen true
Color $005C5C5C
Orientation Landscape
Static false
Active true
Sniff Object Events false
Show Ruler false
Grid
Components
Properties Events
Display: Screen1
Components palette
Basic
Box
Rounded Box
Circle
Line
Label
Button
Rounded Button
Circle Button
Image
CheckBox
Radio Button
ProgressBar
Layers
Show All Layers
Visible Layer name Lock layer
Layer0
Screen1
Welcome
Display: Screen1
User Code
No object selected Screen1 C:\Users\anikolic.MIK\Documents\MyFirstProject.vtft
Total objects on screen: 0



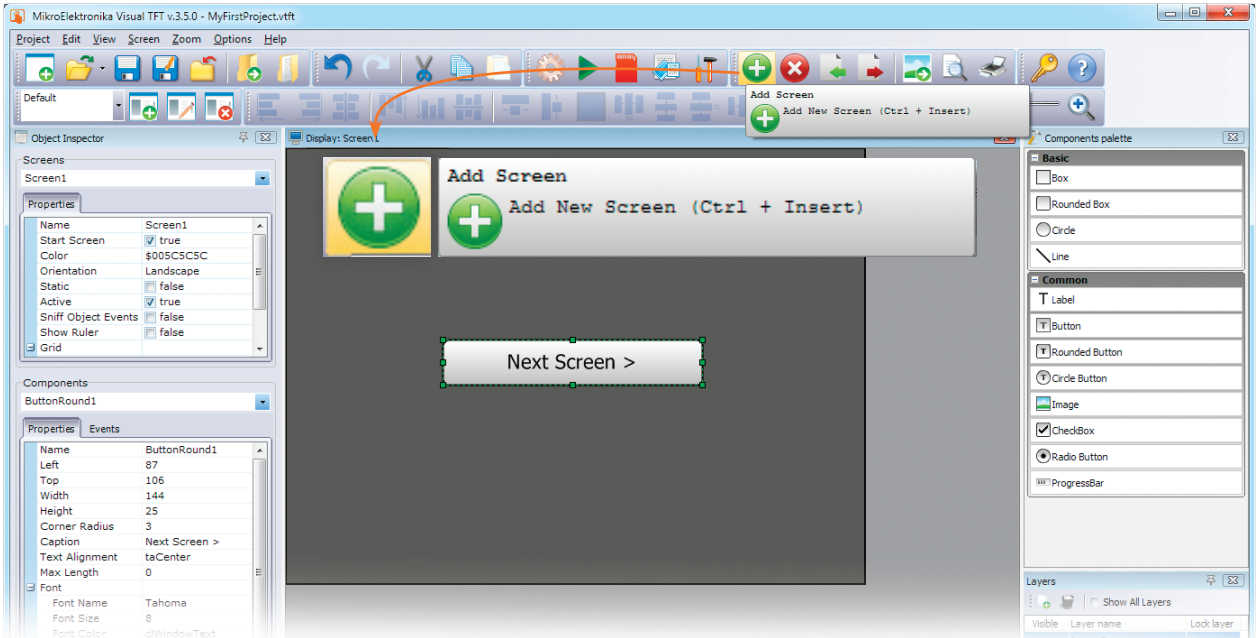
Place a button on the screen

We will start by placing a **Rounded Button** component onto the Screen1. Just click and drag the component from the components palette and drop it over the center of the Screen1.



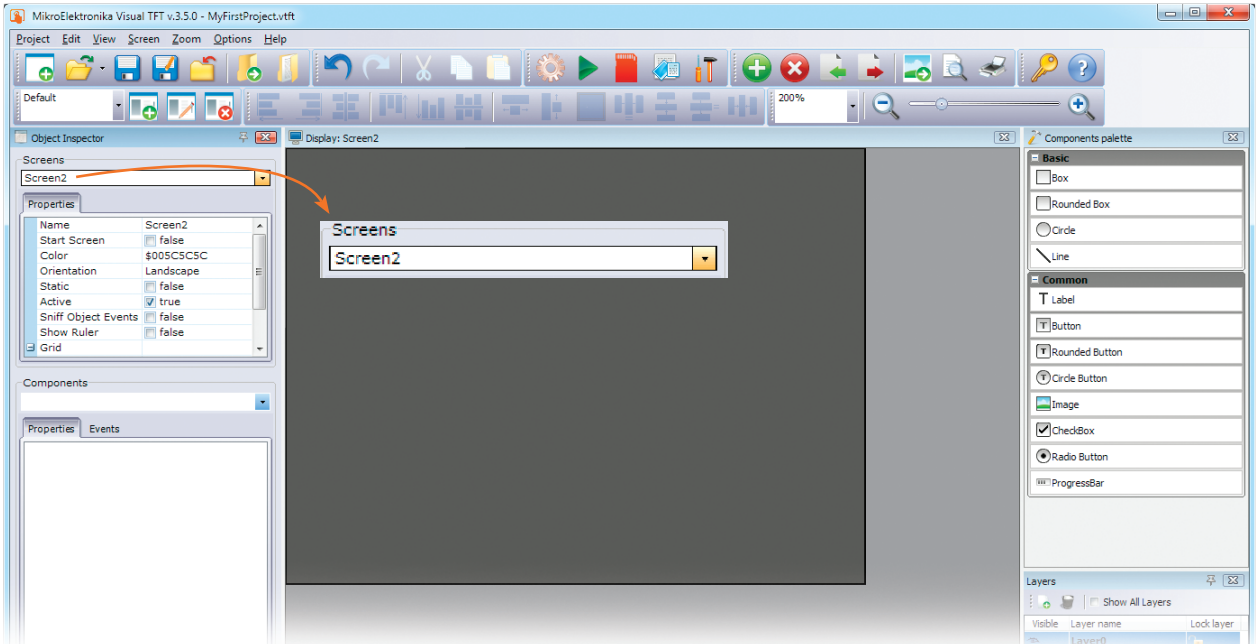
Change button caption to “Next Screen >”

Click the button to select it. It’s properties will be shown in Components section of the **Object Inspector**. Click the **Caption** property and change it to **“Next Screen >”**. Button will be instantly updated. Readjust it’s size and position as shown in the screenshot above.



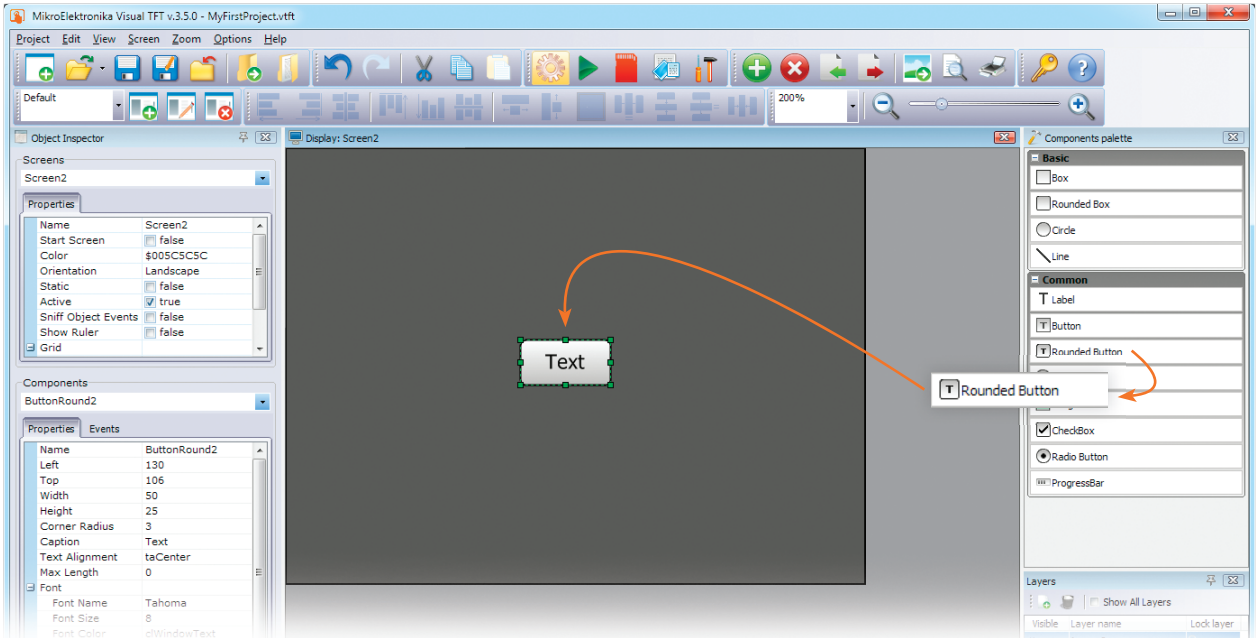
Add another screen using the toolbar button

Let's add another screen now. In the toolbar section locate the green round button with the white "plus" in the middle. It is used for adding screens to the project. Click this button to add a new screen.



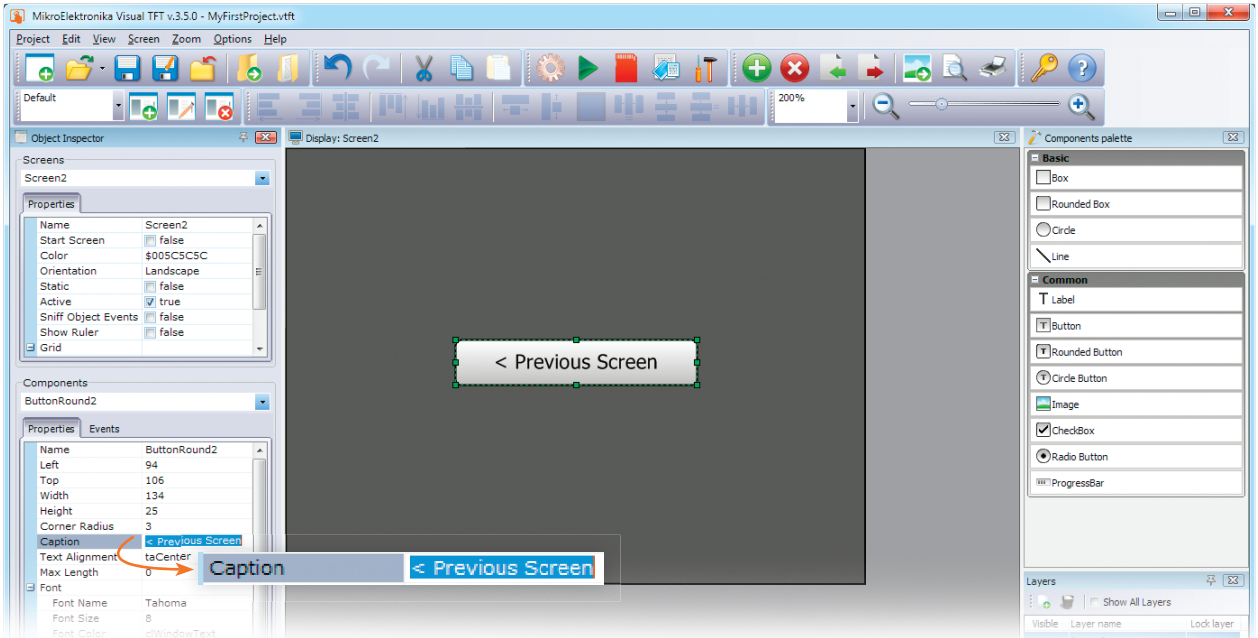
New empty Screen2

New screen is automatically named **Screen2**. It will be initially empty. You can traverse through screens using the dropdown list in the Screens section of the Object Inspector.



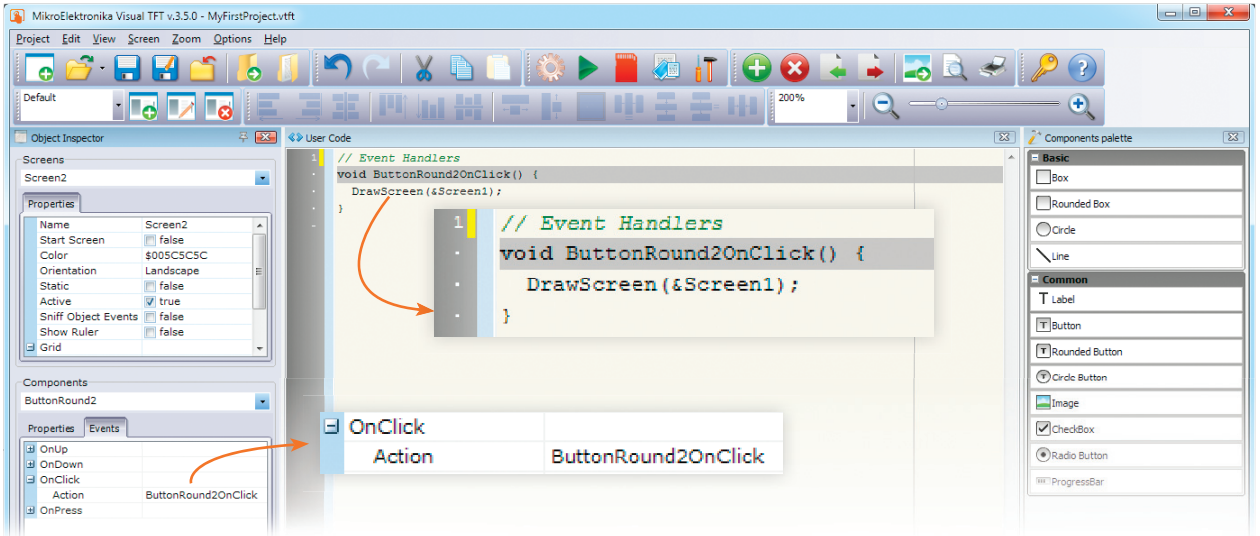
Place a button on Screen2

Let's now place a button on **Screen2**. As in the previous case, just click and drag the **Rounded Button** component from Components Palette window and drop it over the center of the Screen2. A new component named **ButtonRound2** will appear.



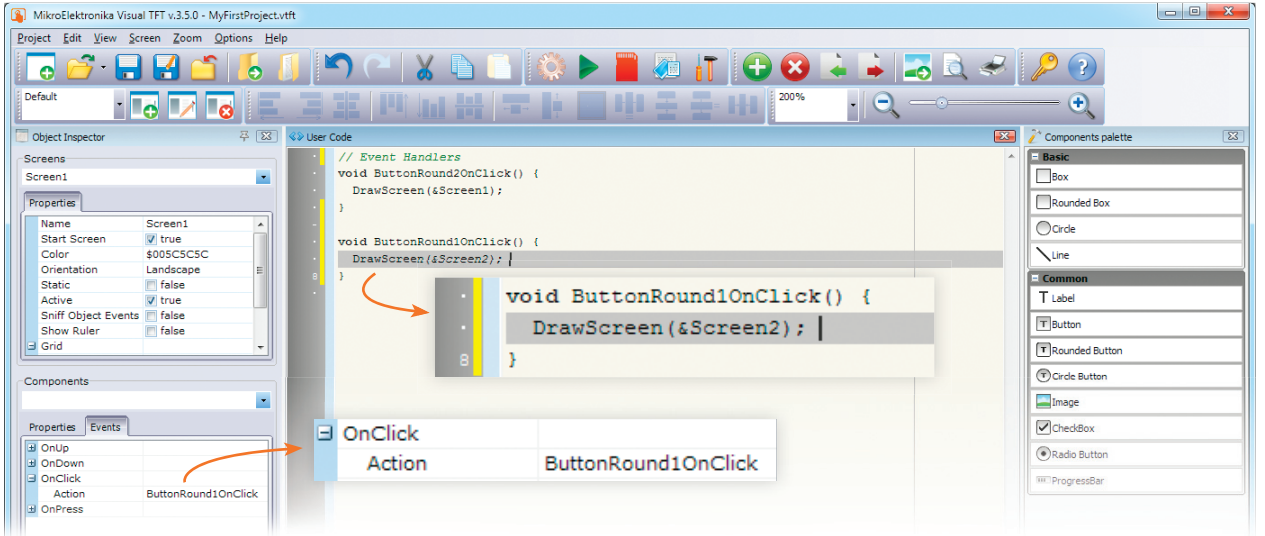
Change button caption to "< Previous Screen"

Click the **ButtonRound2** to select it. It's properties will be shown in Components section of the **Object Inspector**. Click the **Caption** property and change it to **"< Previous Screen"**. The button will be instantly updated.



Add "OnClick" event code to ButtonRound2

It's time to specify the function of the buttons when clicked. In order to do that we will add **OnClick** events to both buttons. Locate the OnClick event of the **ButtonRound2** in the of the Events Tab of **Object Inspector**. Double click it. The **User Code window** will appear. It will contain the function prototype that is automatically associated with the click event. In the function body just type the following line of code: **"DrawScreen(&Screen1);"**. This code will be executed when the button is clicked, thus invoking the drawing of Screen1.



Add "OnClick" event code to ButtonRound1

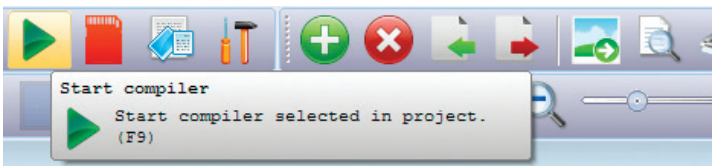
Select **ButtonRound1** component from the dropdown list in the **Components** section of the **Object Inspector** window. Double click it's **OnClick** event in the Events Tab to create and associate the corresponding function. Type the following line of code in the body of the function using the editor of **User Code window**: **"DrawScreen(&Screen2);"**. The code will be executed when the button is clicked, thus invoking the drawing of Screen2. So, when we click the button on the first screen it will take us to the next screen, and when we click the button there it will return us to the initial screen.

5. Building the code in the compiler

We have now successfully created a new project, designed a new graphical interface with two screens and two buttons, and defined their behaviour. All we have to do now is to generate the application code and build it with **mikroC PRO for PIC** compiler, assuming that you have already downloaded and installed the mikroC PRO for PIC compiler and that you have a valid license (USB Dongle or KeyFile License). If not, please visit the compiler website, download the Demo version, and consider purchasing the license:

http://www.mikroe.com/mikroc/pic/

In **Visual TFT toolbar**, locate the **Start Compiler** button with the green triangle. We call it the Play button. When clicked it will automatically generate code for the target compiler, and launch the compiler with the project loaded, as shown in the screenshot on **Figure 5.1**.



The entire code is ready to be built as soon as the compiler is started. No additional interventions are required. We can initiate project building using **Build->Build [CTRL+F9]** command. After the compilation and linking is done successfully, the message window should contain this information, as shown in the screenshot below.

Line	Message No.	Message Text	Unit
0	127	All files Compiled in 203 ms	
0	1144	Used RAM (bytes): 526 (14%) Free RAM (bytes): 3357 (86%)	Used RAM (bytes): 526 (14%) Free RAM (bytes): 3357 (86%)
0	1144	Used ROM (bytes): 25032 (19%) Free ROM (bytes): 106032 (81%)	Used ROM (bytes): 25032 (19%) Free ROM (bytes): 106032 (81%)
0	125	Project Linked Successfully	MyFirstProject.mcppi
0	128	Linked in 187 ms	
0	129	Project 'MyFirstProject.mcppi' completed: 562 ms	
0	103	Finished successfully: 19 Feb 2013, 15:47:16	MyFirstProject.mcppi

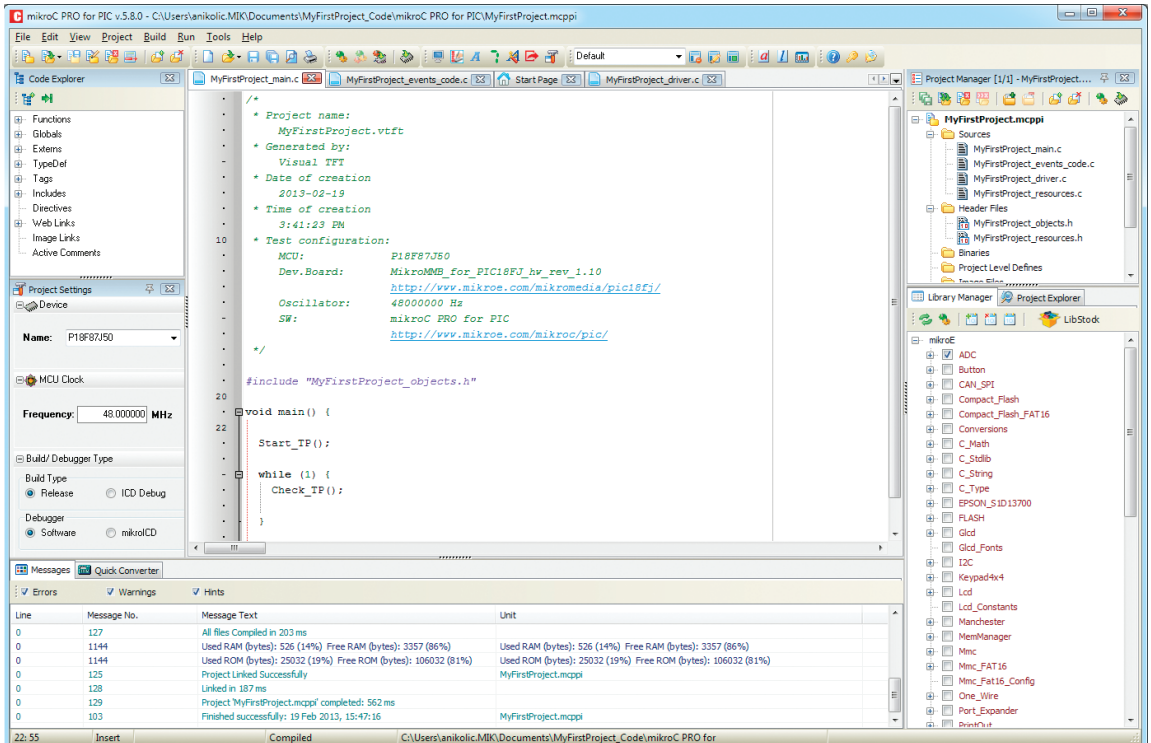


Figure 5-1: mikroC PRO for PIC compiler loaded with our first Visual TFT project

6. Uploading the firmware to MCU

After the project is built, the compiler produces a .HEX file which can be downloaded into the target microcontroller on the **mikromedia for PIC18FJ** development board. Programming of the MCU is done using external **mikroProg for PIC, dsPIC and PIC32** programmer/debugger and the software called **mikroProg Suite for PIC**. Software and programmer drivers are usually installed together with the compiler. Prior to programming, make sure that mikroProg is connected to your PC via USB programmer connector, and that USB and LINK LEDs are active. In order to initiate programming just hit **[F11]** button in the compiler.

Figure 6-1: mikromedia board connected to mikroProg programmer

7. Test on target hardware

When programming is done, the application will start and the Graphic Display will show calibration screen. This is due to the piece of code automatically created by Visual TFT software. It will help you to calibrate the Touch panel using 2-point calibration procedure. After that, the initial screen will appear. It features the **"Next Screen >"** button, exactly as we have intended. One click of the button takes us to the next screen. Click of the **"< Previous Screen"** button brings us back to the first screen.

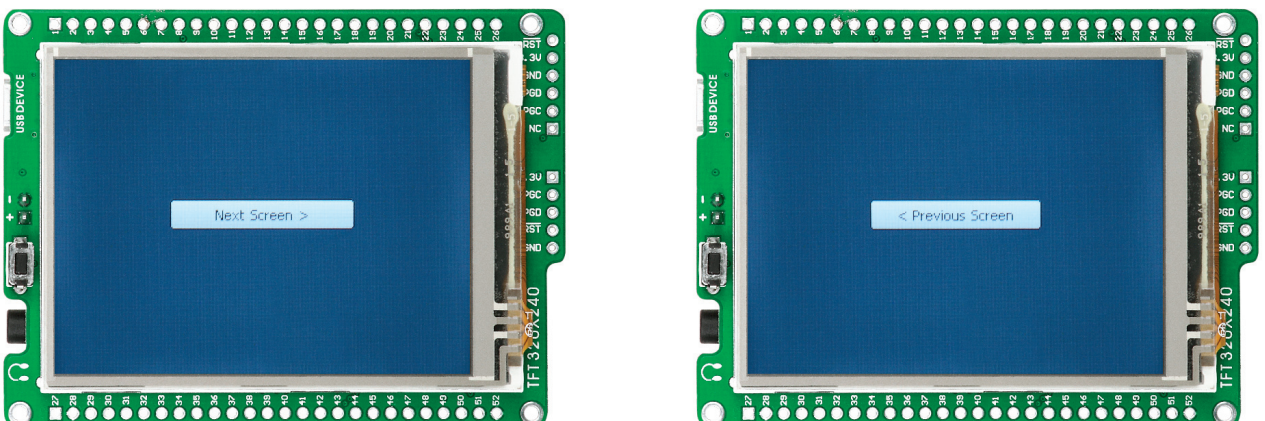


Figure 7-1: Your first project's appearance on target hardware

8. Learn more with provided examples

Voila! You first project is up and running. You must be excited to see those buttons doing exactly what you excpected. It's now time to move on and get to know other Visual TFT features as well. There's no better way to explore all the possibilities of the software than to experiment with provided example projects. Examples like the **Calculator** show how to use matrix of customized buttons to create

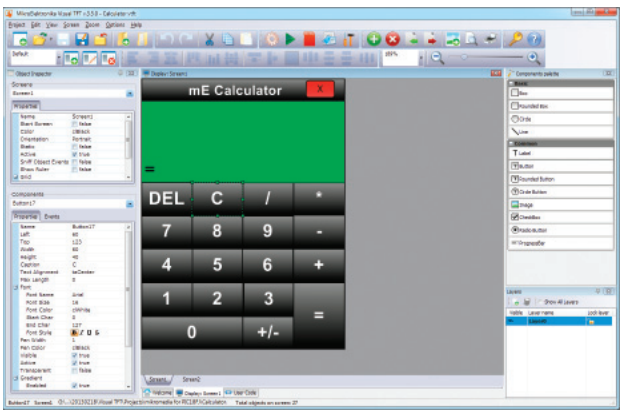


Figure 8-1: Calculator Demo

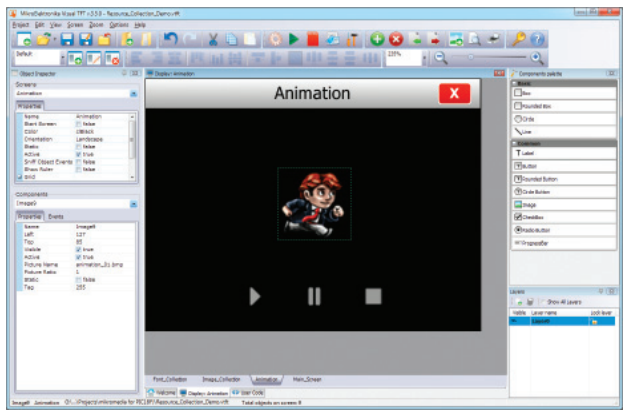


Figure 8-2: Resource Collection Demo

your own touch-screen calculator. **Resource Collection Demo** covers the topic of using fonts and images stored in the resource file on your MMC/SD card to create image sliders, animations, etc. **Progress Bar Demo** shows the possibilities of ProgressBar, CheckBox and RadioButton components. **Simple Maze** is a demonstration of how to make simple gaming user interfaces.

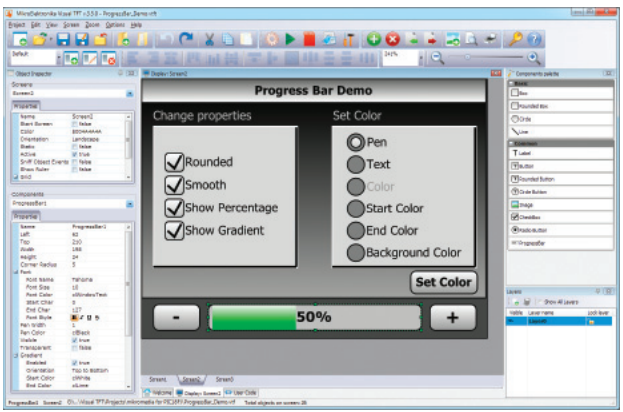


Figure 8-3: Progress Bar Demo

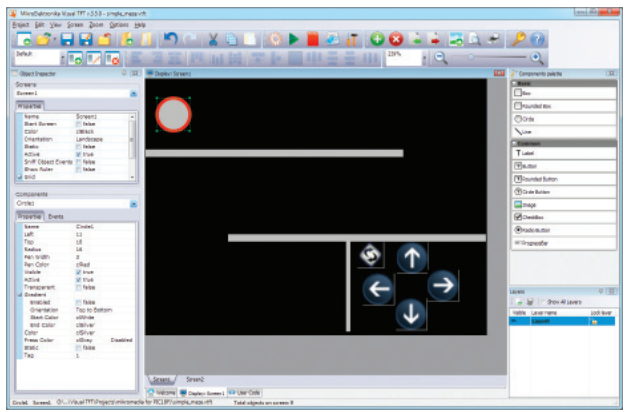


Figure 8-4: Simple Maze Demo

9. Support for FT800/EVE controller from FTDI

A wide variety of industry sectors are now demanding more sophisticated forms of human-machine interaction and expecting more satisfying user experiences. This is why MikroElektronika teamed up with FTDI chip® company to provide development support for their latest **EVE GUI Platform** and **FT800** family of graphics controllers. EVE integrates display, audio and touch onto a low cost, easy-to-use, single-chip solution.

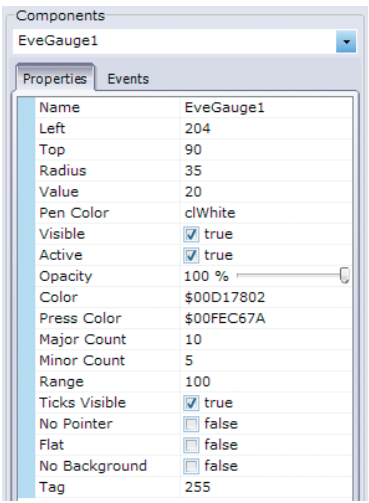


Figure 9-1: EveGauge component and it's set of supported features

The EVE family has an object based structure (where objects can be images, fonts, etc) that presents engineers with an easy way to design more effective GUIs for TFTs, with all the display, audio and touch functionality included.

Visual TFT is the first software in the world to provide full support for many of EVE's powerful features like sound, transparency and anti-aliasing fonts. There are 12 new components available for GUI design, which are natively supported in the controller itself.

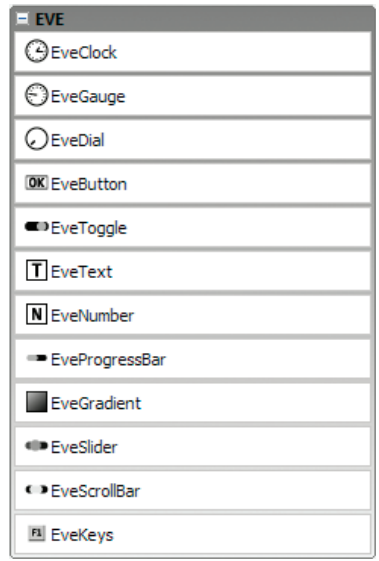


Figure 9-2: An Additional set of 12 EVE components supported natively by the controller

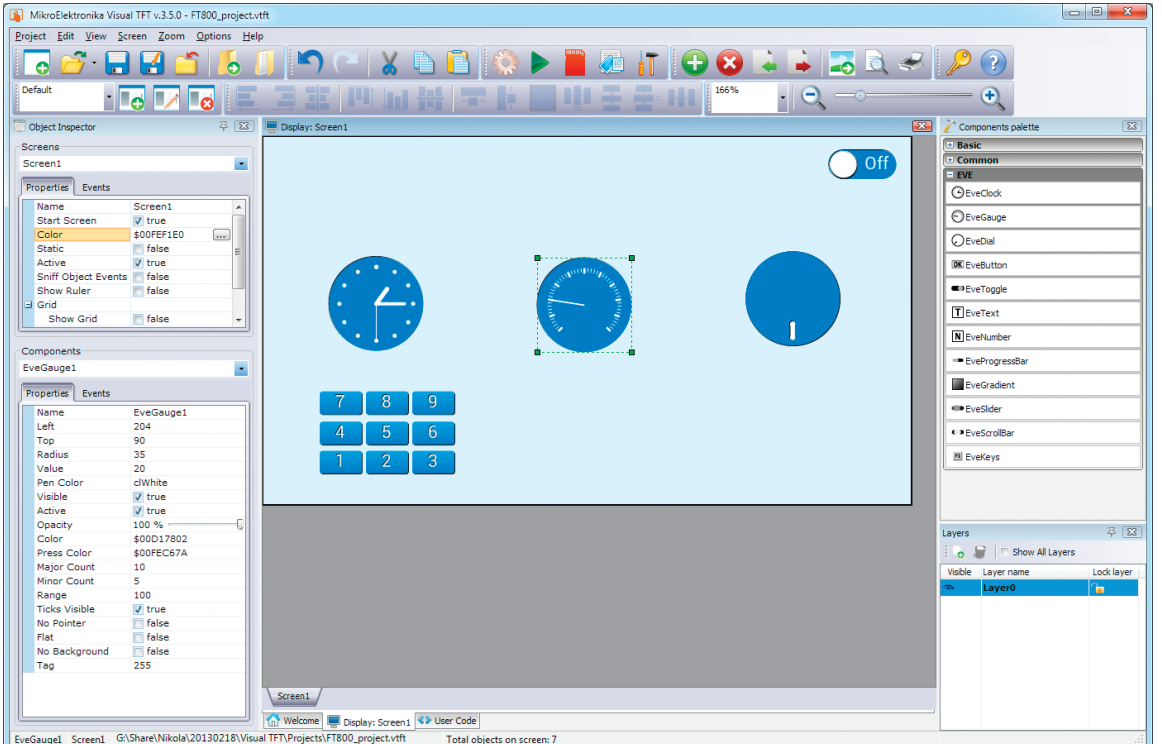


Figure 9-3: Visual TFT Project Workspace with EVE components on Screen

10. What's Next?

We have successfully created together your first project in **Visual TFT** software. But this is only the first step. You can now continue on your own, but will always have our help and support along the way.

Projects

Choose the development board and compiler and you are ready to start writing your projects. We have equipped **Visual TFT** with dozens of examples that demonstrate the use of every single feature of the software. There are interesting examples for each supported board. They are an excellent starting point for your future projects. Just load the example, read well commented code, and see how it works on hardware. Click the **Open example** button located on the welcome screen to browse through the Projects folder:

\Visual TFT\Projects

Community

If you want to find answers to your questions on many interesting topics we invite you to visit our forum at **http://www.mikroe.com/forum** and browse through more than 185 thousand posts. You are likely to find just the right information for you. On the other hand, if you want to download free projects and libraries, or share your own code, please visit the Libstock™ website. With user profiles, you can get to know other programmers, and subscribe to receive notifications on their code.

http://www.libstock.com/

Support

We all know how important to have someone to rely on in moments when we are stuck with our projects, facing a deadline, or when we just want to ask a simple, basic question, that's pulling us back for a while. We do understand how important this is and therefore our Support Department is one of the pillars upon which our company is based. MikroElektronika offers Free Tech Support to the end of product lifetime, so if something goes wrong, we are ready and willing to help!

http://www.mikroe.com/esupport/

DISCLAIMER

All the products owned by MikroElektronika are protected by copyright law and international copyright treaty. Therefore, this manual is to be treated as any other copyright material. No part of this manual, including product and software described herein, may be reproduced, stored in a retrieval system, translated or transmitted in any form or by any means, without the prior written permission of MikroElektronika. The manual PDF edition can be printed for private or local use, but not for distribution. Any modification of this manual is prohibited. MikroElektronika provides this manual 'as is' without warranty of any kind, either expressed or implied, including, but not limited to, the implied warranties or conditions of merchantability or fitness for a particular purpose.

MikroElektronika shall assume no responsibility or liability for any errors, omissions and inaccuracies that may appear in this manual. In no event shall MikroElektronika, its directors, officers, employees or distributors be liable for any indirect, specific, incidental or consequential damages (including damages for loss of business profits and business information, business interruption or any other pecuniary loss) arising out of the use of this manual or product, even if MikroElektronika has been advised of the possibility of such damages. MikroElektronika reserves the right to change information contained in this manual at any time without prior notice, if necessary.

HIGH RISK ACTIVITIES

The products of MikroElektronika are not fault - tolerant nor designed, manufactured or intended for use or resale as on - line control equipment in hazardous environments requiring fail - safe performance, such as in the operation of nuclear facilities, aircraft navigation or communication systems, air traffic control, direct life support machines or weapons systems in which the failure of Software could lead directly to death, personal injury or severe physical or environmental damage ('High Risk Activities'). MikroElektronika and its suppliers specifically disclaim any expressed or implied warranty of fitness for High Risk Activities.

TRADEMARKS

The MikroElektronika name and logo, the MikroElektronika logo, mikroC™, mikroBasic™, mikroPascal™, mikroProg™, mikromedia™ and Visual TFT™ are trademarks of MikroElektronika. All other trademarks mentioned herein are property of their respective companies. All other product and corporate names appearing in this manual may or may not be registered trademarks or copyrights of their respective companies, and are only used for identification or explanation and to the owners' benefit, with no intent to infringe.