Embedded Wizard

Workshop

Date: December 4th 2017

Embedded Wizard is a product of **TARA** Systems

Agenda

- Introduction
- Embedded Wizard
  - Architecture
  - Platform Package
  - Embedded Wizard Studio, the IDE
  - IDE Details
  - Mosaic Framework
  - Sample GUIs
  - FAQ
- Interactive Hands-On-Training

Embedded Wizard

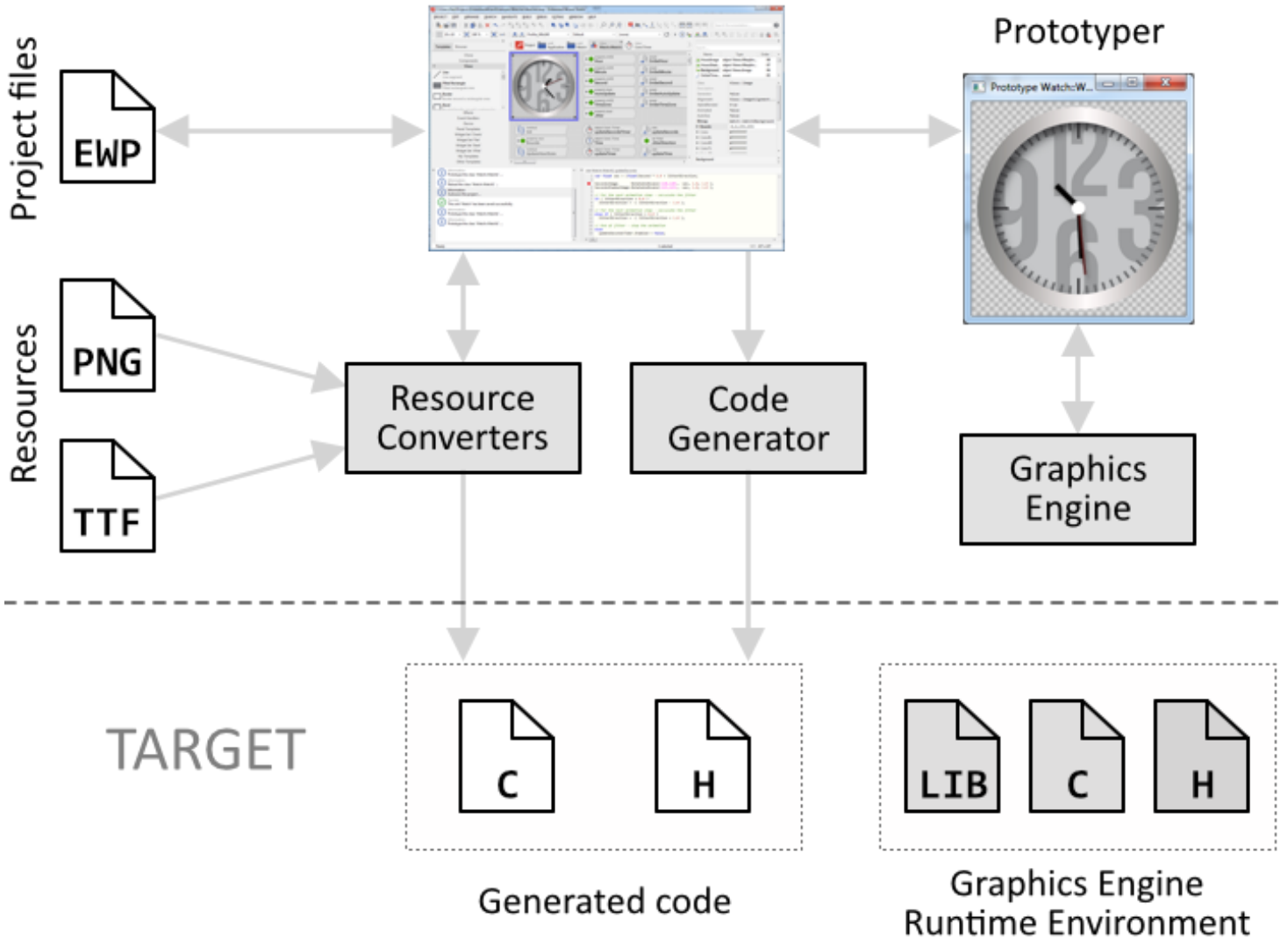
Architecture

Architecture

Embedded Wizard

Platform Package

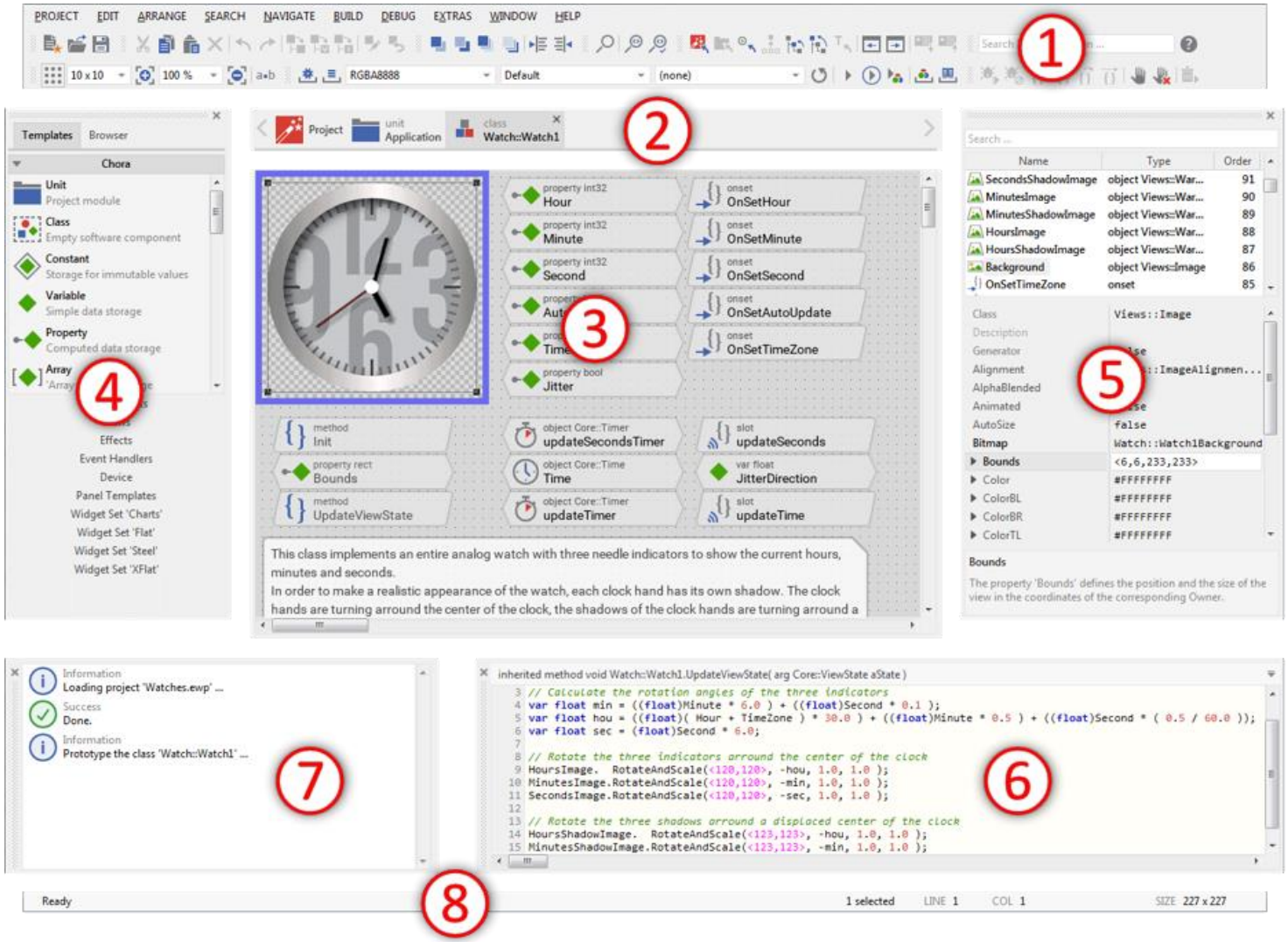
Platform Package



Embedded Wizard

Embedded Wizard Studio, the IDE

Embedded Wizard Studio - the IDE



1. Menu and Toolbar
2. Navigation Bar
3. Composer, Bricks
4. Templates and Browser
5. Inspector
6. Code Editor
7. Log Window
8. Status Bar

Embedded Wizard

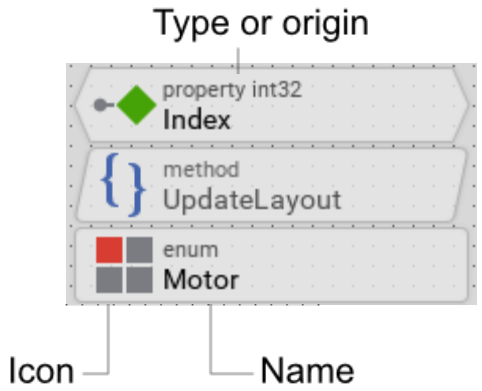
Embedded Wizard

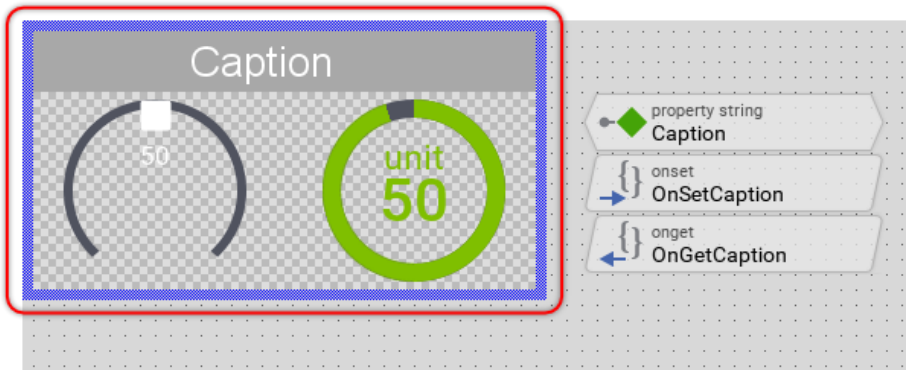
IDE Details

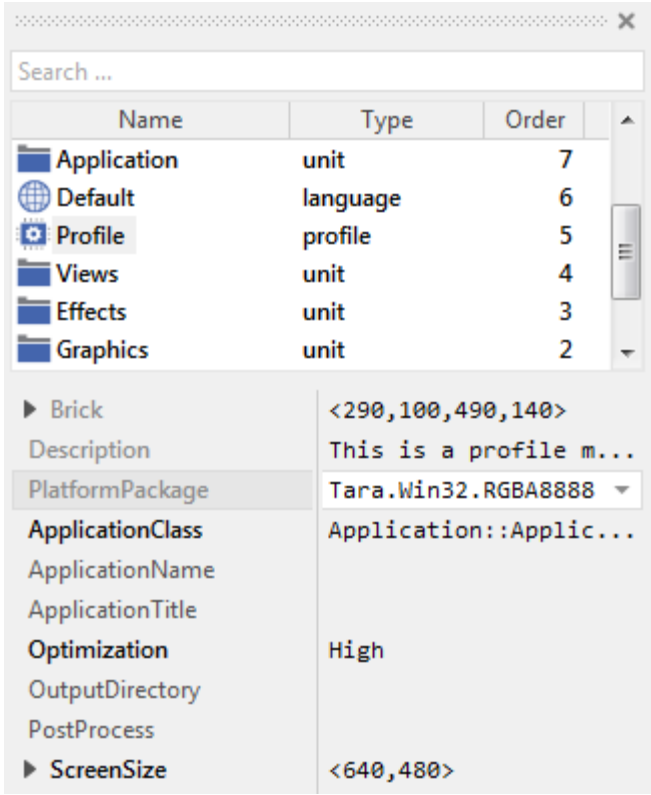
IDE Details

Composer

- Is used to build up the components of a GUI application, to arrange and connect all members in a visual way (WYSIWYG).
- Bricks: representation of logic or data (methods, variables, units ...)
- Project view: profile, language, style, unit, framework, application
- Bounds: selection and positioning of views, drag & drop, copy & paste, undo & redo

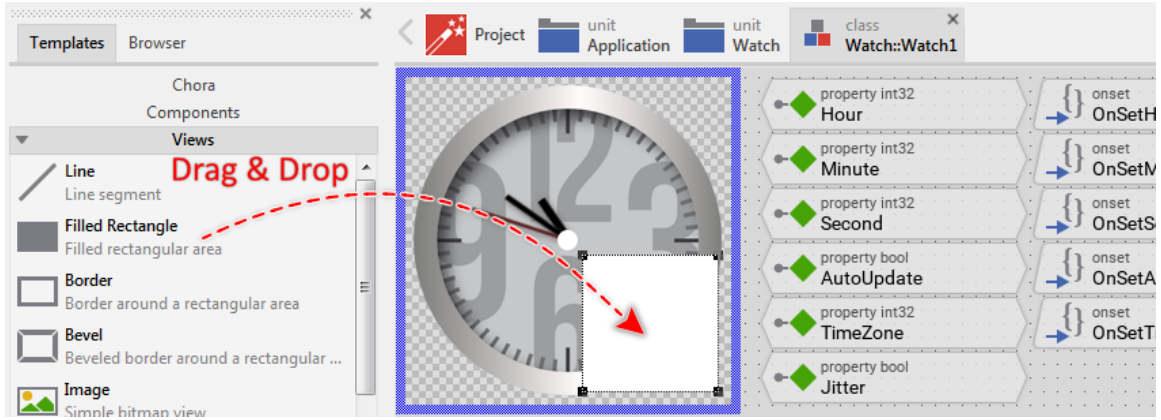


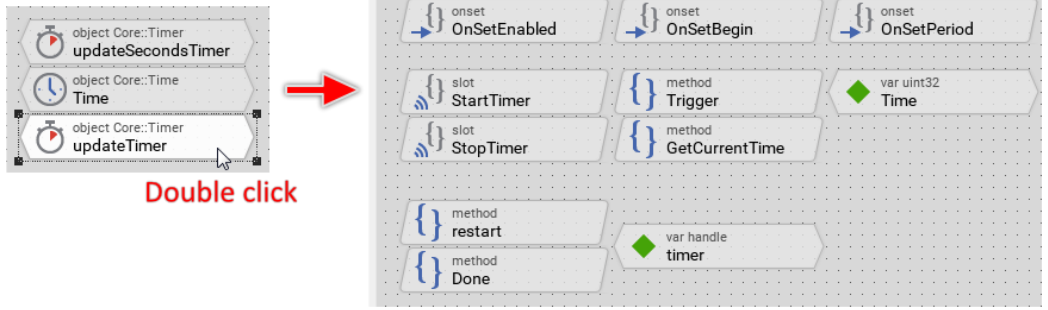




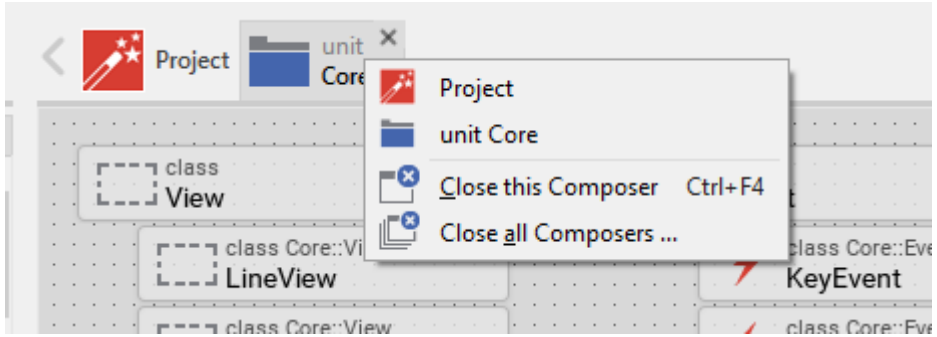
IDE Details

Composer





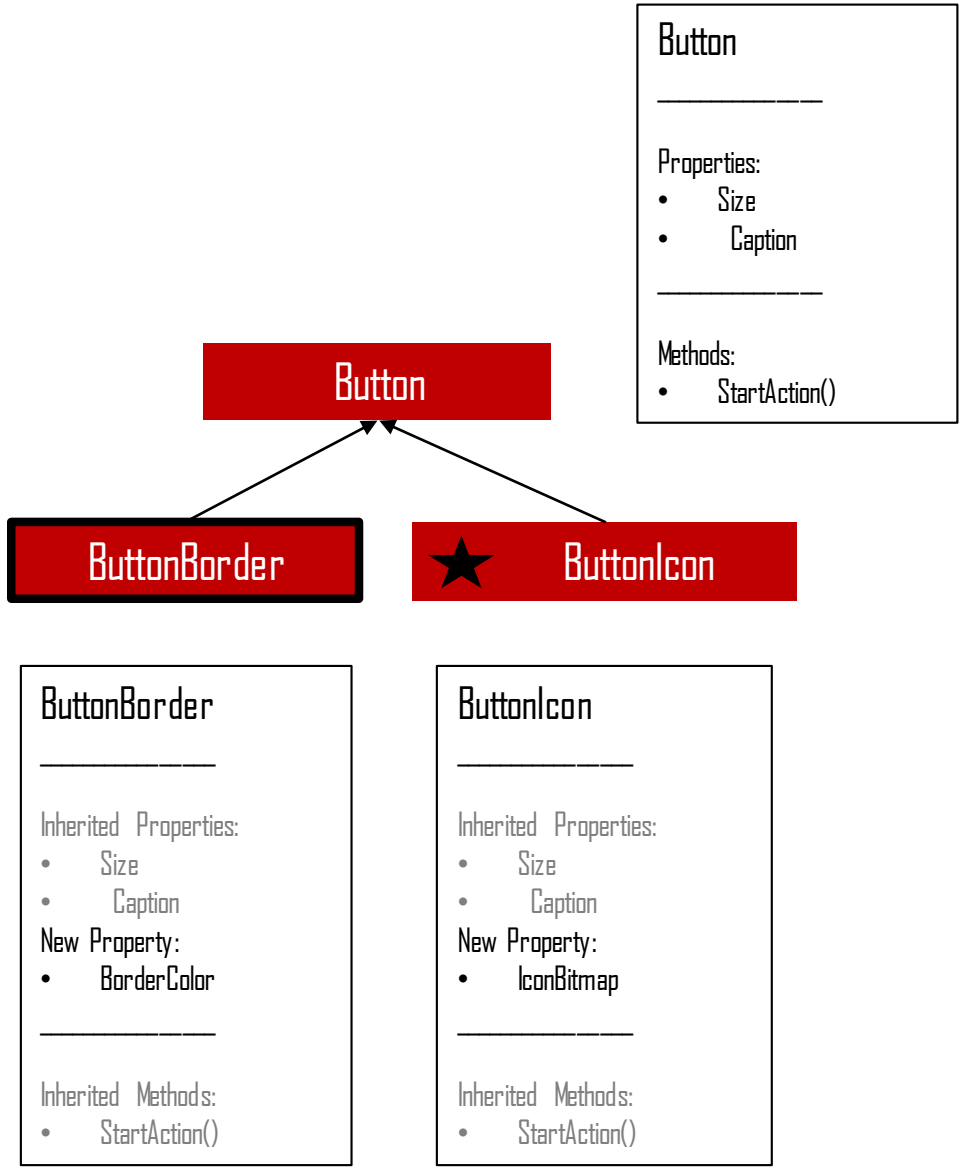
Navigation



Excursus: OOP

Object oriented Programming

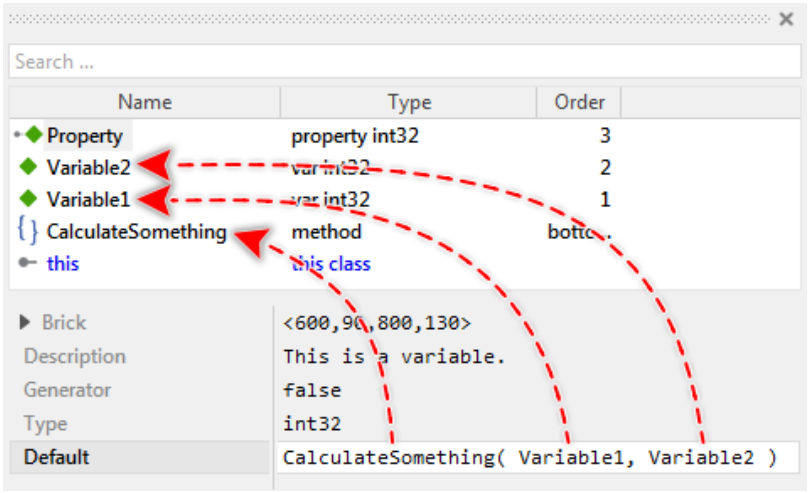
- Objects, Classes, Inheritance, Derivation
- Split complex applications into independent/reusable "objects" that contain certain properties
- Access properties of an object programmatically,
  e.g. ButtonBorder.Caption = "Start";
- Classes can be derived from other classes (new class inherits all properties/functions from super class) and is able to get extended
- Try to find similarities between objects
- Optimizes reusability, maintenance and extension

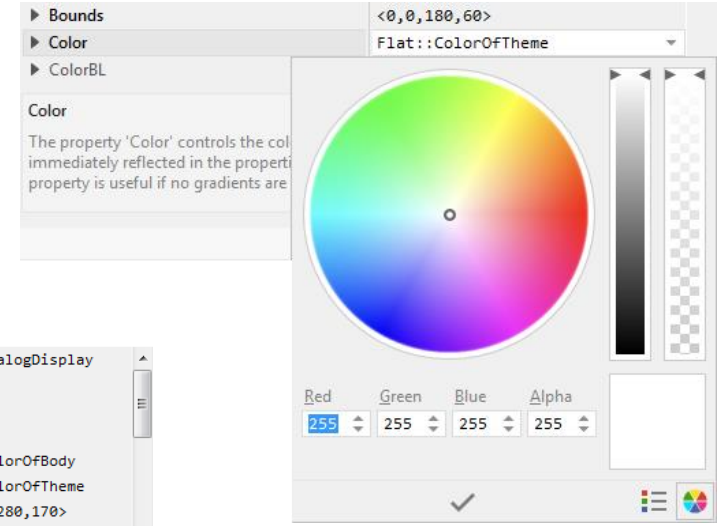


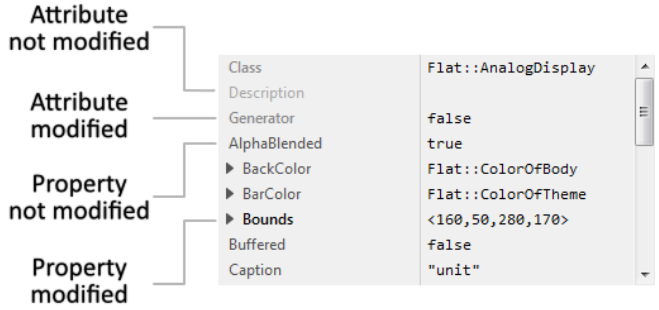
IDE Details

Inspector

- Is used to inspect and modify the members and their attributes or properties.
- Top: Viewing of all members
  - objects/variables, spelling and word formatting (bold ...), renaming, z-order, restack
- Bottom: Viewing and modifying attributes and properties of the members
  - Attribute window with class, description, corresponding properties like color, text, bounds ..., restoring values, assistants, word formatting (bold ...), usage of constants
  - Description field



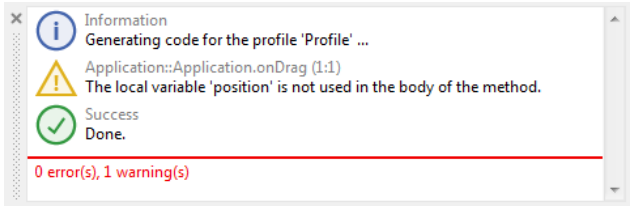




IDE Details

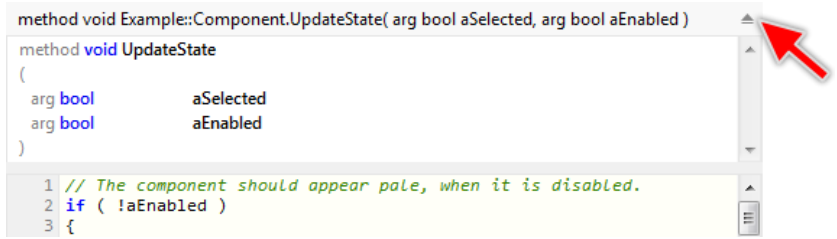
Log Window

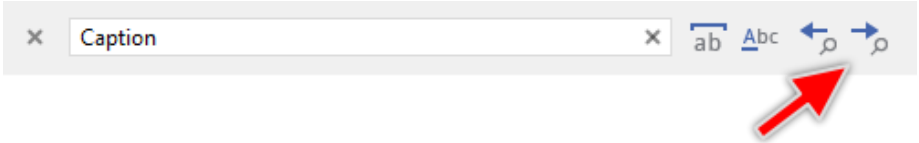
- Displays all information, warnings, errors and custom traces.



Editor

- Is used to implement the desired behavior of your GUI components in Chora.
- Trace, hotkeys (Ctrl + space, Ctrl + f ...), context menu, comments
- Init/Done: constructor/destructor most time derivable, otherwise creatable
- Properties: Getter, setter (see Chora)
- Methods: Header with arguments and return value optionally
- Slots: Methods with benefits (see Chora).



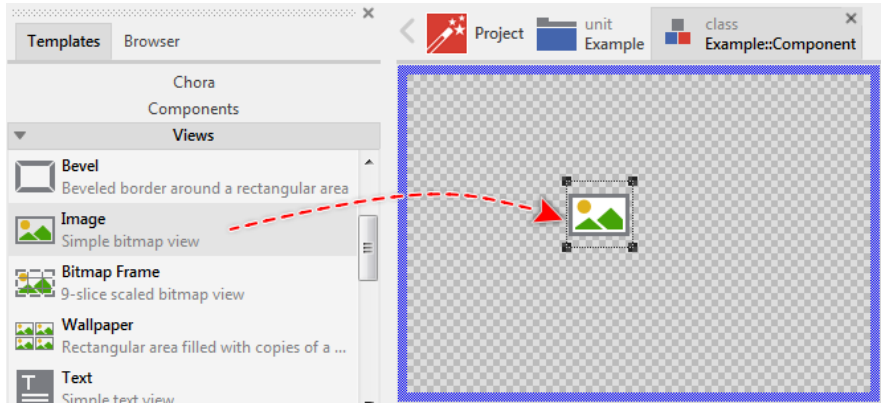


Embedded Wizard

IDE Details

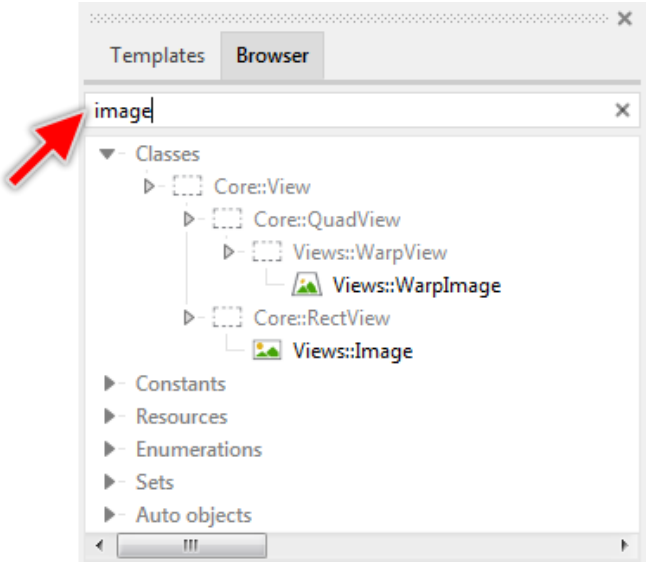
Gallery-Templates

- Includes Chora language types
- Includes predefined templates
- Includes ready-to-use widgets
- Elements are to be dragged into the Composer view



Gallery-Browser

- Overview about all project members (e.g. classes, constants, enums, resources ...) and their hierarchy



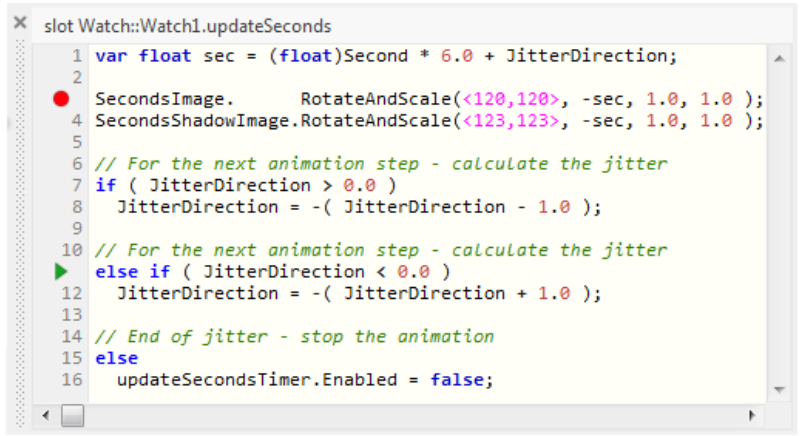
IDE Details

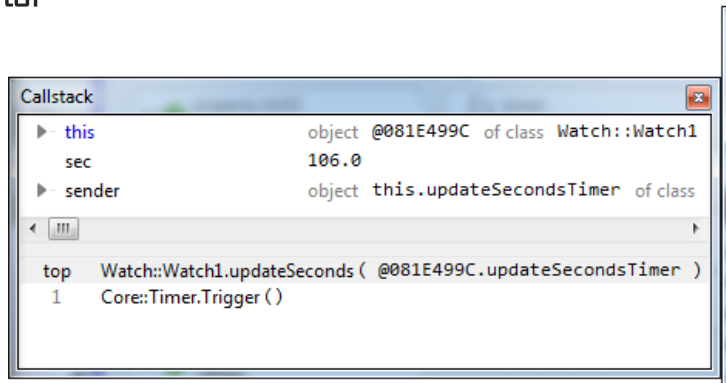
Build

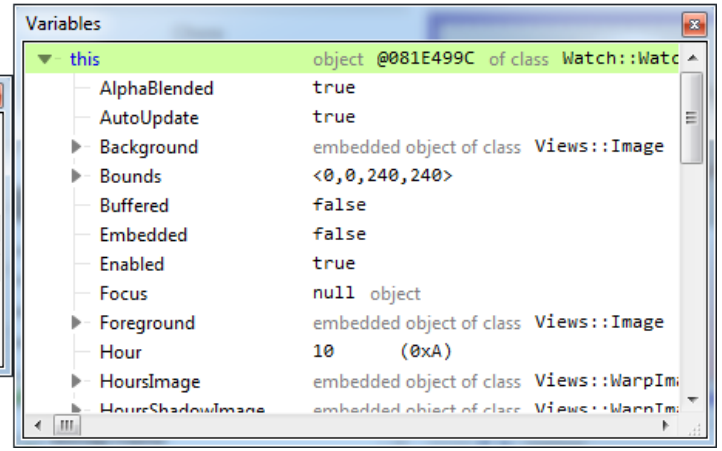
- Code generator (build profile or in batch mode)
- Ewu, generated Code, Split
- Documentation generator

Debug

- Prototyper: Enables instant testing of the look & feel
- Windows: Callstack, Chora objects, garbage collector
- Breakpoints and stepping
- Track screen updates
- Traces
- Debugging via Composer







Embedded Wizard

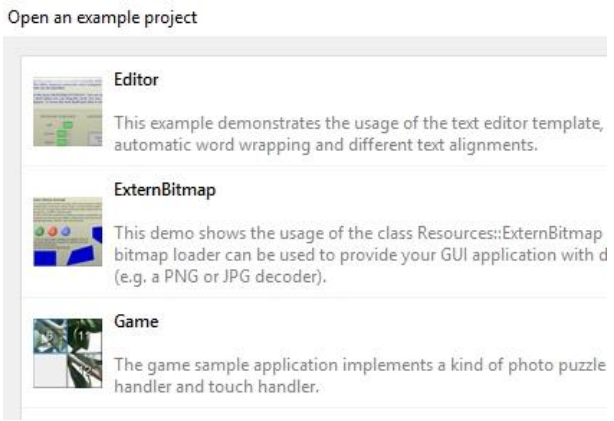
IDE Details

Work with Embedded Wizard

- Read the Basic Concepts of the tool: doc.embedded-wizard.de/basic-concepts
- Install Embedded Wizard Studio and the Platform Package
- Except the Free Edition, the Hardware Dongle is necessary

Help

- Context sensitive help in IDE (focus and press F1)
- Tutorials on YouTube: youtube.com/c/EmbeddedWizard
- Online documentation: doc.embedded-wizard.de
- Ask Embedded Wizard Forum: ask.embedded-wizard.de
- Use Samples and Widgets to learn about certain aspects ("Don't invent the wheel!")



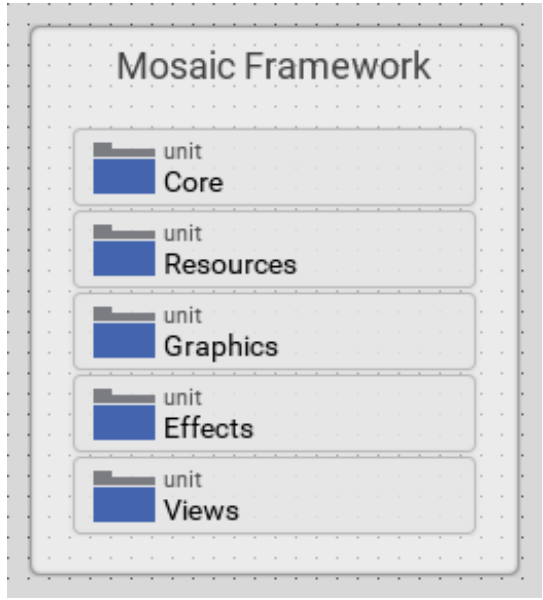
Embedded Wizard

Embedded Wizard

Mosaic Framework

Mosaic Framework

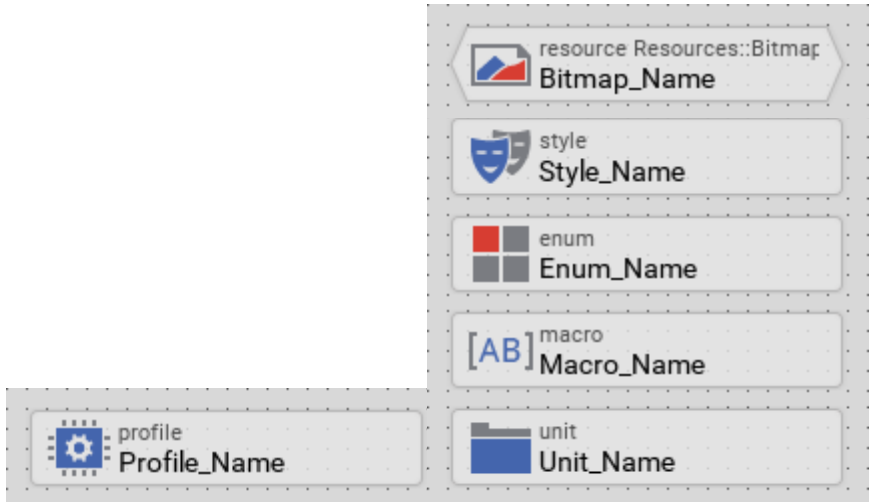
- Each Embedded Wizard project is based on this framework
- It implements the core functionality of a GUI
- Mosaic Framework based on Chora programming language
- Most of the custom classes are derived from Core::Group
- The Graphics provides the fundamental classes needed to perform graphical operations
- Use the Effects to enrich your GUI with smooth animations
- The Views provide a base implementation for the most common GUI elements



Excursus: Chora

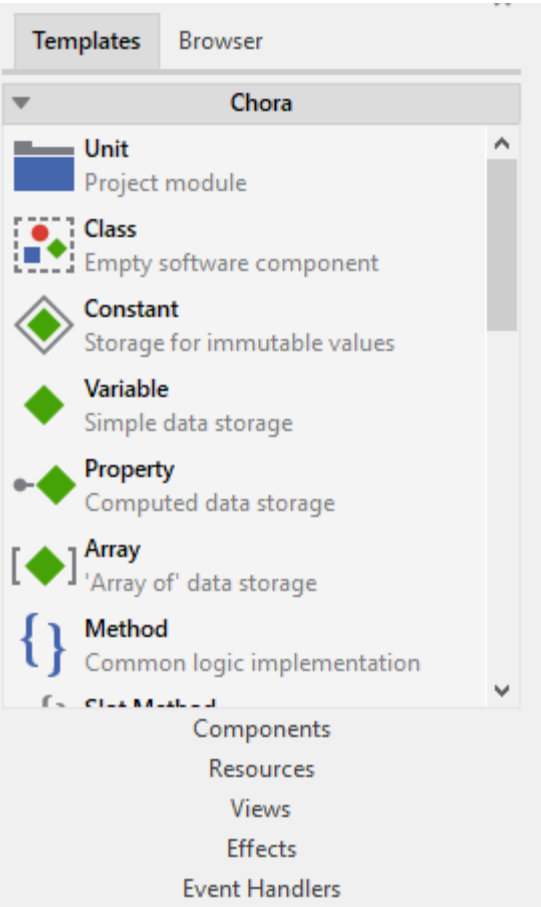
Chora Programming Language - why an own language?

- Optimized for GUI development (e.g. new data types like language)
- Fully object orientated with integrated garbage collection
- Built-in support for MVC, MVP & MVVM design patterns
- Integrated documentation generator
- Prototyping of whole GUI with own debugger
- Abstraction: platform independent => code generating for ANSI-C and Java Script
- Structure and syntax similar to C, C++ and Java
- See: http://doc.embedded-wizard.de/chora-reference



Mosaic Framework: Chora in detail

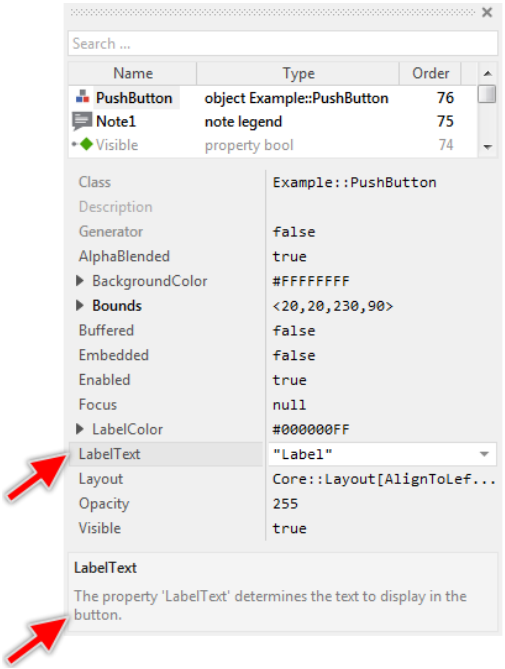
- **Chora:** Unit, Class, Constant, Variable (types: int, float, color, rect, point, language ...), Property (see next slide), Array, Method, Slot (see following slides), Enum, Set, Language, Style, Profile, Macro, Inline Code, Annotation
  - Object oriented programming: super class, derive class, modify object via editor (example: rectangle.color = #FF0000FF;)
  - Built-ins: var, if, switch, ~~break,~~ for, while, return, attach-/detachobserver, ...
- **Components**: Application, Component and Templates for widgets
- **Resources:** Bitmap, ExternBitmap, Font, Attributed Set
- **Views:** Line, Rectangles, Image, Wallpaper, Text, Group, Warp, Lists, OutlineBox
- **Effects:** Timer (default, bool, int32, float, color, point, rect)
- **Event Handlers:** Key-/Touch-Handler (Touch: simple, slide, wipe, rotate)

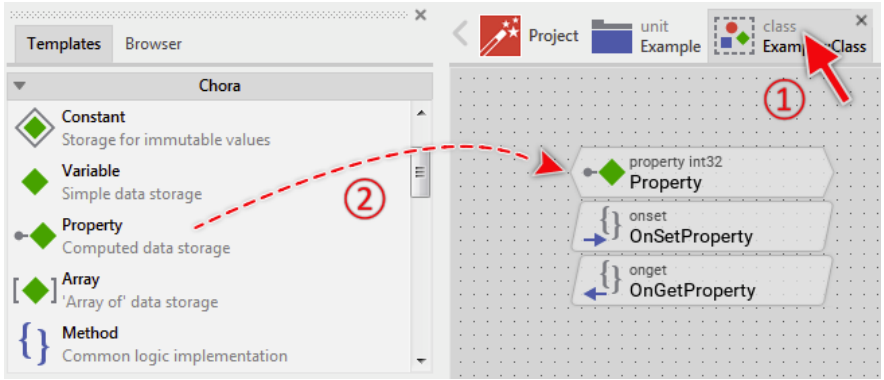


Excursus: Property

The better variable

- Whenever a property is evaluated or modified, its getter and setter are executed:
  - OnSet: is called when the property is about to be changed
  - OnGet: is called when the property is read
- Pure: To write a new value to the Property, you need to prefix it
- Value: The (hidden) argument that contains the new value to assign
- Property reference: ^Property delivers the reference of the Property
- Value of a Property reference: Property^
- All properties that are added to a component are automatically listed in the Inspector window

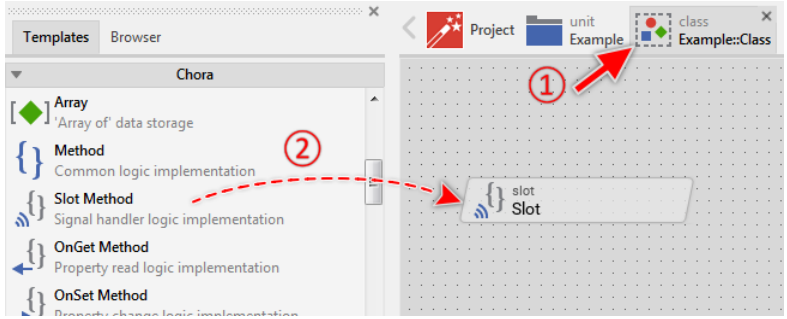


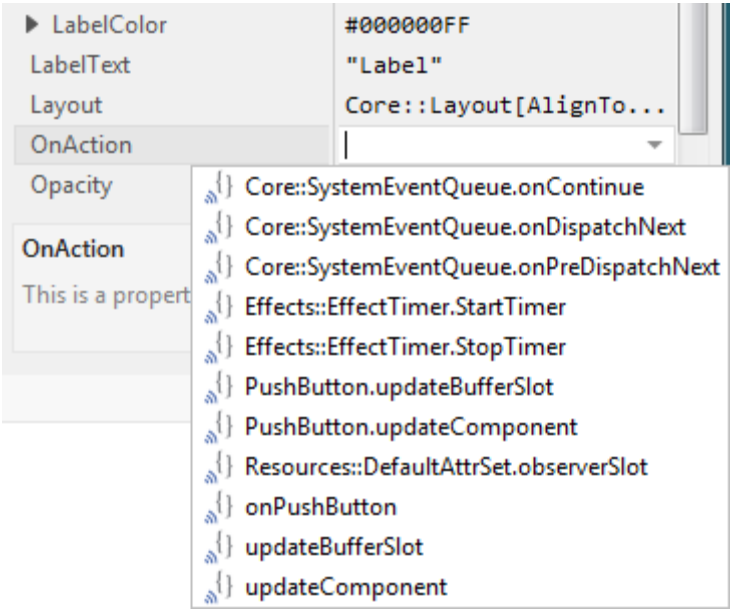


Excursus: Slot

The well-timed method

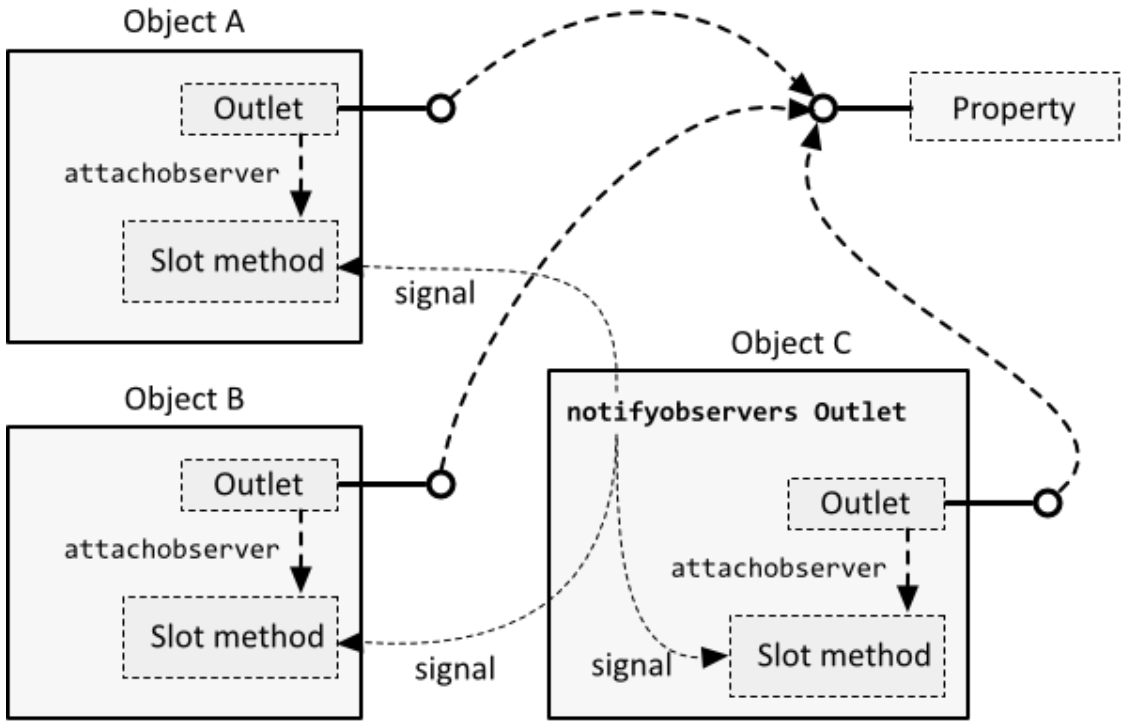
- No return value, no arguments, but able to time:
  - Postsignal: execution time is before next screen update, only once
  - Signal: immediate, synchronous call of slot method
  - Idlesignal: execution time is after postsignals, only once
  - => e.g. postsignal onUpdate;
- Sender
  - Optional slot parameter, which refers to the object that has sent the signal
- It is very convenient to use Properties declared with slot data type
  - Example: "OnAction" Property of a button





Excursus: Notifications and Observer

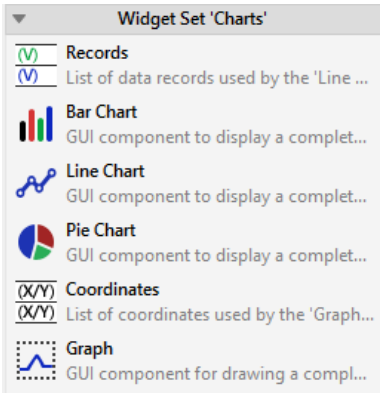
- Attach and detach an Observer to a reference
  - Example:
    attachobserver Slot, ^Property;
    means that "Slot" will be executed, when ^Property get notified.
- Notify all Observers to a reference
  - Example:
    notifyobservers ^Property;
    means to notify all observers that are attached to the reference.
- Slots are executed "postsignal"
- This pattern is already used by Outlets or Property Observer.
- This technique is following the model-view-controller (MVC) programming pattern.

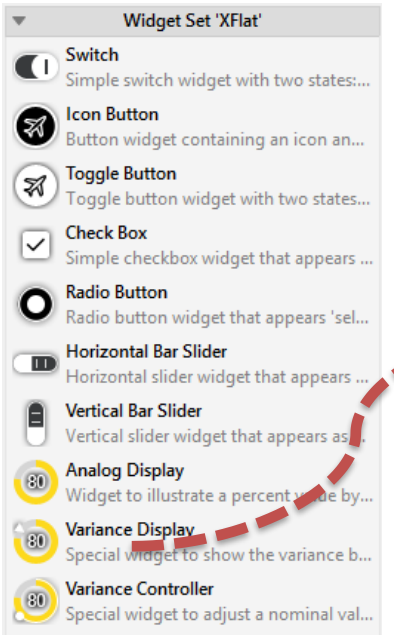


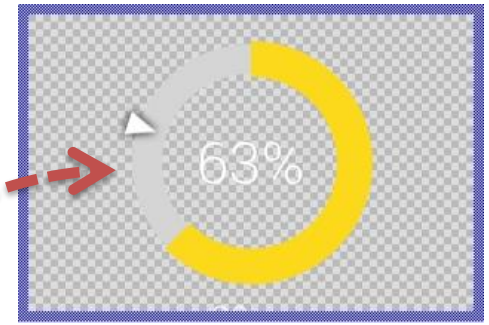
Embedded Wizard

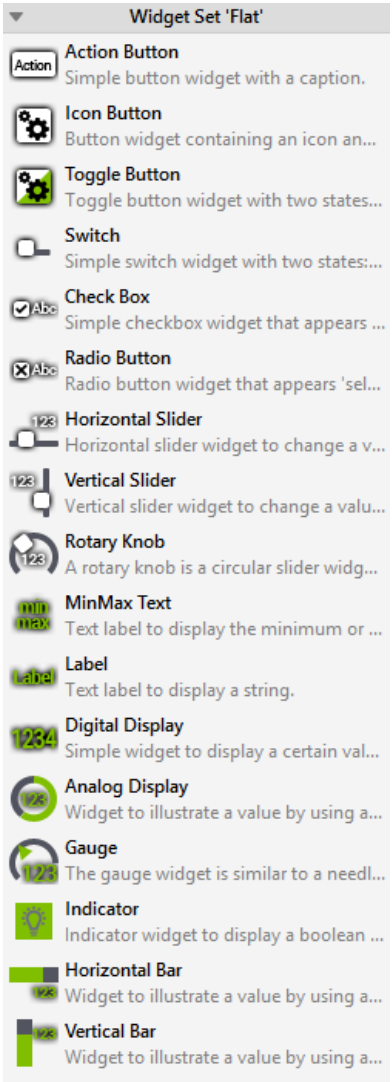
Excursus: Widgets

- Graphical components for standard user interactions (e.g. Slider)
- Ready to use Widgets included
- Extendable with Custom Widgets
- Drag and Drop from Gallery
- Easy to use for a first click dummy
- Re-usable
- Extendable







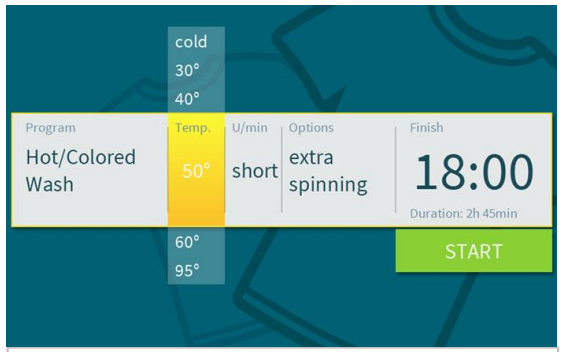


Embedded Wizard

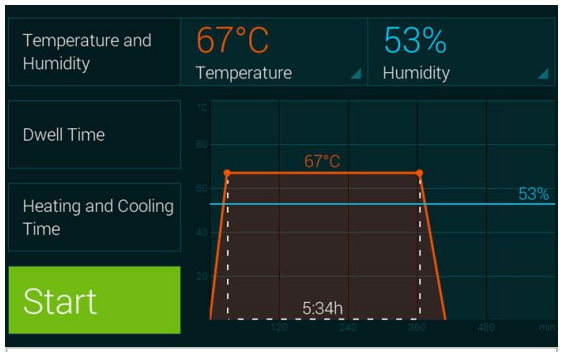
Embedded Wizard

Sample GUIs

Sample GUIs



Washing Machine

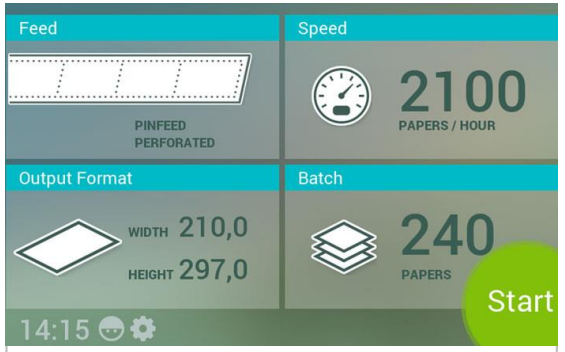


Climatic Cabinet

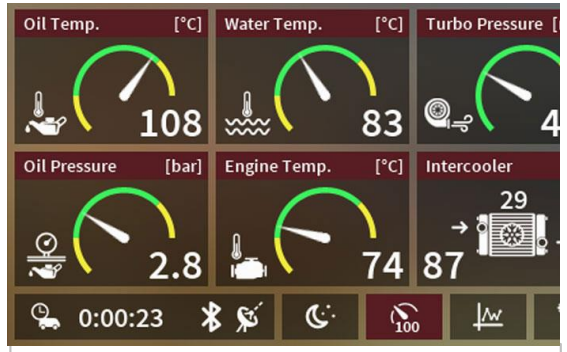
Various Gauges

Various Charts

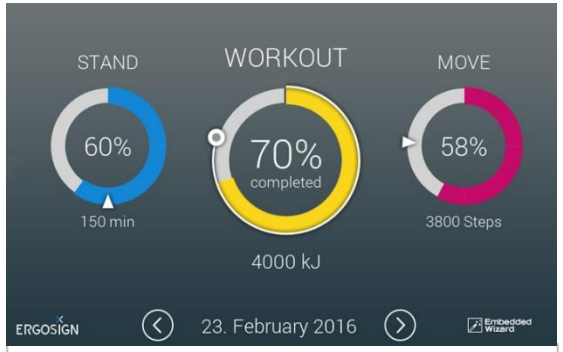
Smart Thermostat



Paper Cutting Machine



Vehicle Data Logger



Fitness Tracker

Embedded Wizard

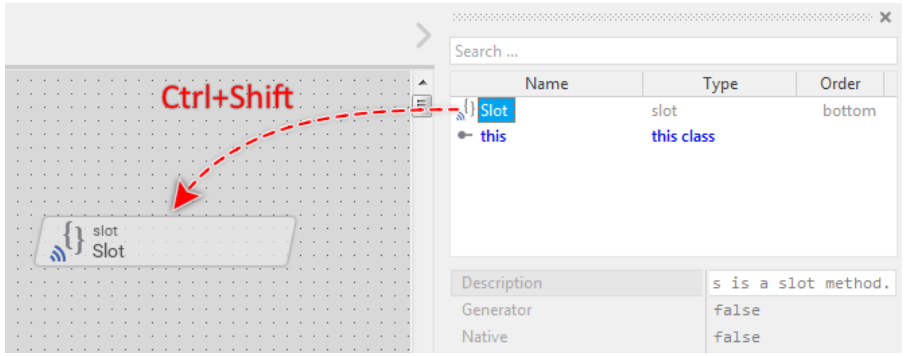
Embedded Wizard

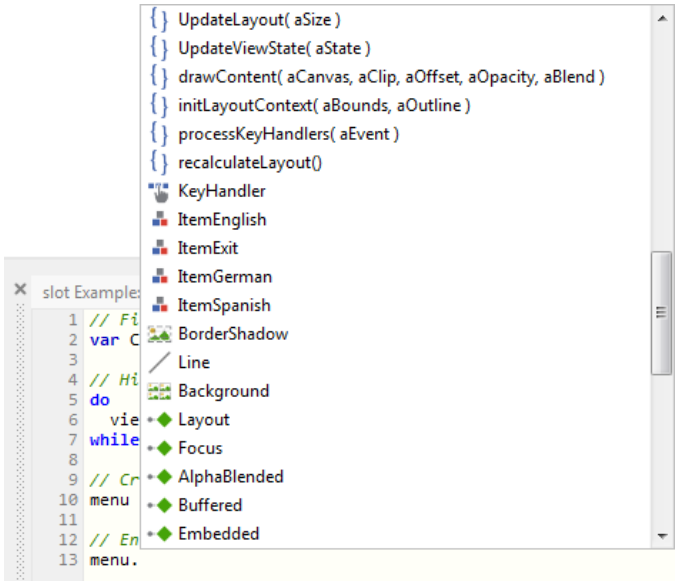
Frequently Asked Questions (FAQ)

FAQ: Hotkeys

Hotkeys

- Renaming: F2
- Restack, Inspector: Ctrl + up or + down
- Override, Inspector: Ctrl + Shift + drag & drop, Ctrl + m
- Breakpoint in Editor: F9
- Start Prototyper: F5 or Ctrl + F5 (entire application)
- Help: F1
- Global search: Ctrl + Shift + f
- Create new instance of a brick: Ctrl + Alt + drag & drop
- Editor search/replace: Ctrl + f, Ctrl + h





FAQ: Key Features

- Comfortable IDE with drag & drop
- Visual programming (WYSIWYG) and instant prototyping of UI look & feel
- Simple programming model incl. object-oriented programming support, generating ANSI C and JavaScript
- Platform independent implementation of GUI logic
- Ready-to-use widgets as templates for state-of-the-art designs, including effects (rotation, scaling & perspective transformation each with Hi- and Low-Quality), animations, layout functions, etc.
- (Multi-)Touch, Gestures, Mouse, Remote Control support
- No (RT)OS (i.e. tasks, semaphores, etc.) is required, GUIs can run on bare metal
- UNICODE based
- Supporting various color depths/formats: RGBA8888, RGB888, RGBA4444, RGB565, Index8, LumA44

Links

- Showcases and demos
  www.embedded-wizard.de/demo
- Evaluation Edition
  www.embedded-wizard.de/tryout
- Open Community Support Forum
  ask.embedded-wizard.de
- Online Knowledge Base
  doc.embedded-wizard.de
- Platform Integration – getting started with STM32, NXP, WebGL, Raspberry Pi etc.
  doc.embedded-wizard.de/platform-integration-and-build-environments
- YouTube Channel
  www.youtube.com/c/EmbeddedWizard
- Follow us on Twitter
  www.twitter.com/EmbeddedWizard

Embedded Wizard