TouchGFX

life.augmented

TouchGFX

TouchGFX


LINAK


sodastream
MIX

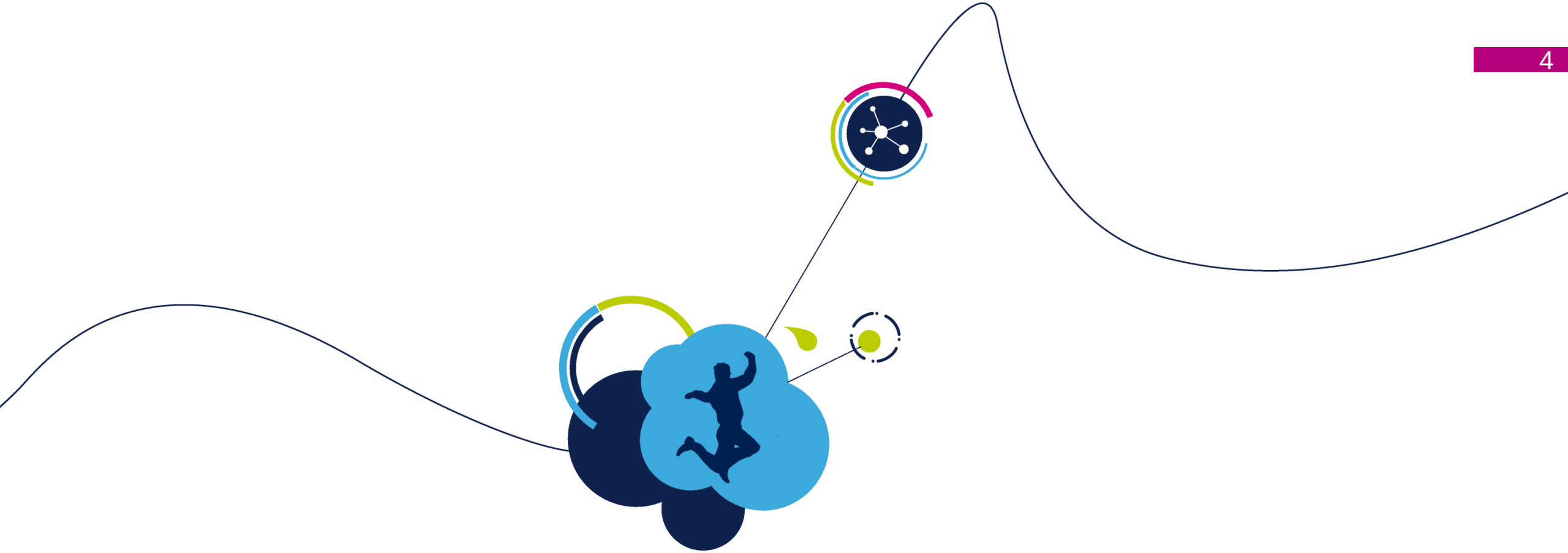

somfy


TEMPERATURE LIVE
LIVING ROOM 20°
MASTER BEDROOM 21°
KITCHEN 22°
HALL 23°
HOME OFFICE 24°
SCHEDULE
AUTO
OFF
MASTER BEDROOM
21°


COPENHAGEN 07:18 SUNDAY 24 NOV
12°
MON 23°
TUE 16°
WED 24°
THU 18°
FRI 17°

Agenda

- What is TouchGFX
- References
- STM32 & TouchGFX
- TouchGFX technical overview
- The TouchGFX framework



What is TouchGFX

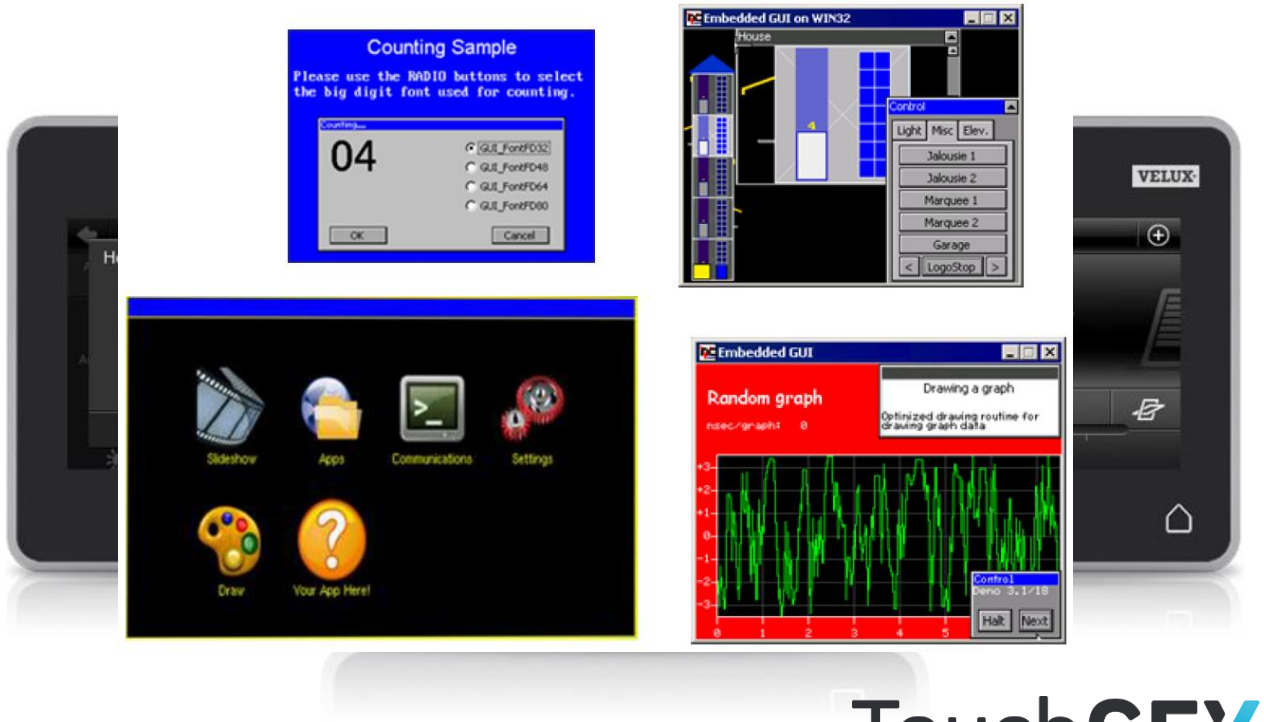
User expectations are high

- Smartphones have become the paramount reference when we judge user interfaces and touch displays, making users of embedded interfaces more demanding

- **The users expect**
  - Touch Gestures
  - Instant Response
  - Intuitive Interaction
  - Modern Design
  - Strong Brand Identity

Typical touch controller interface

TouchGFX enables that



TouchGFX technology

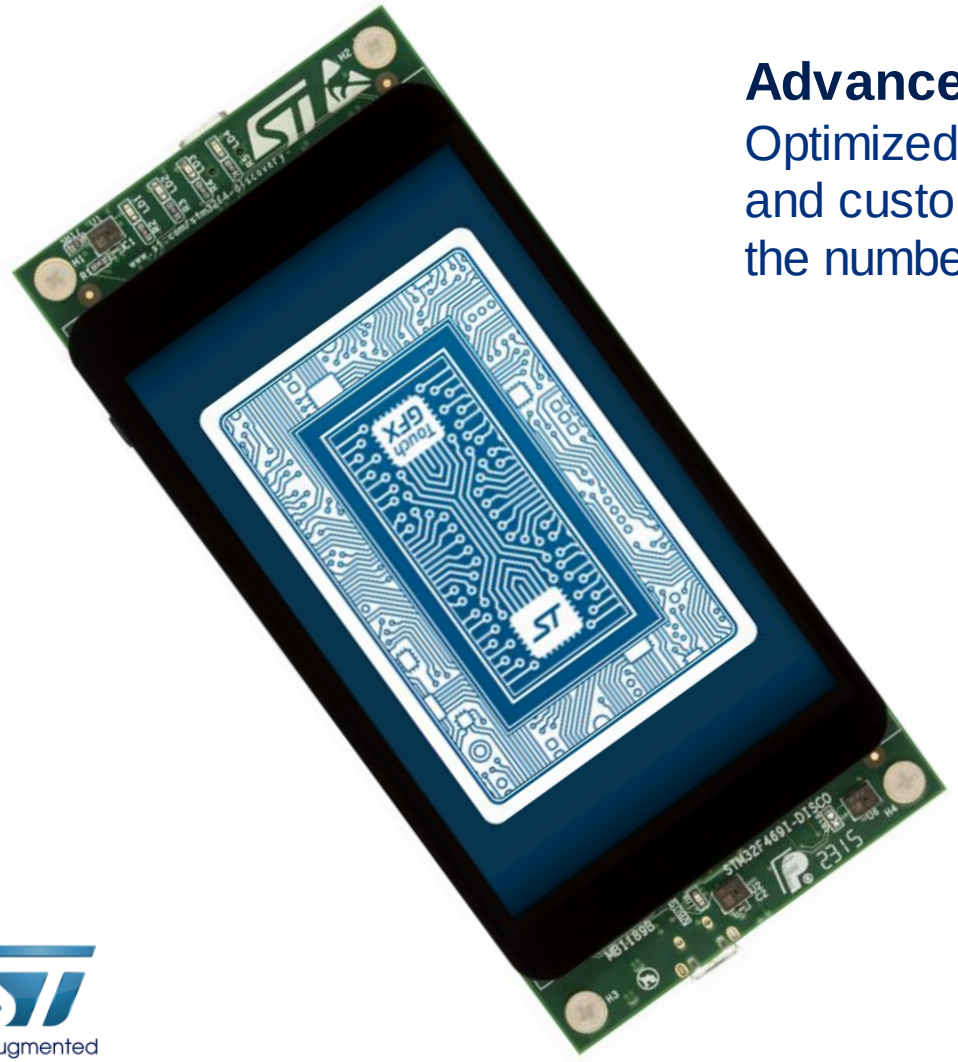
Key Concepts

Optimized for microcontrollers
Highly optimized for performance, flash size, and memory consumption matching the constraints of microcontrollers.

STM32 hardware acceleration
Conversion and analysis of images at compile time ensures optimal use of Chrom-ART accelerator™ features when drawing solid images, images with alpha channel and texts, hereby offloading the MCU.
TouchGFX also utilizes JPEG HW accelerator to ensure fast decoding of JPEG images and M-JPEG video.

TouchGFX technology

Key Concepts



Advanced Rendering Algorithms

Optimized visible surface determination algorithm and customized invalidation techniques minimize the number of drawn pixels.

Easy Creation of Custom Controls

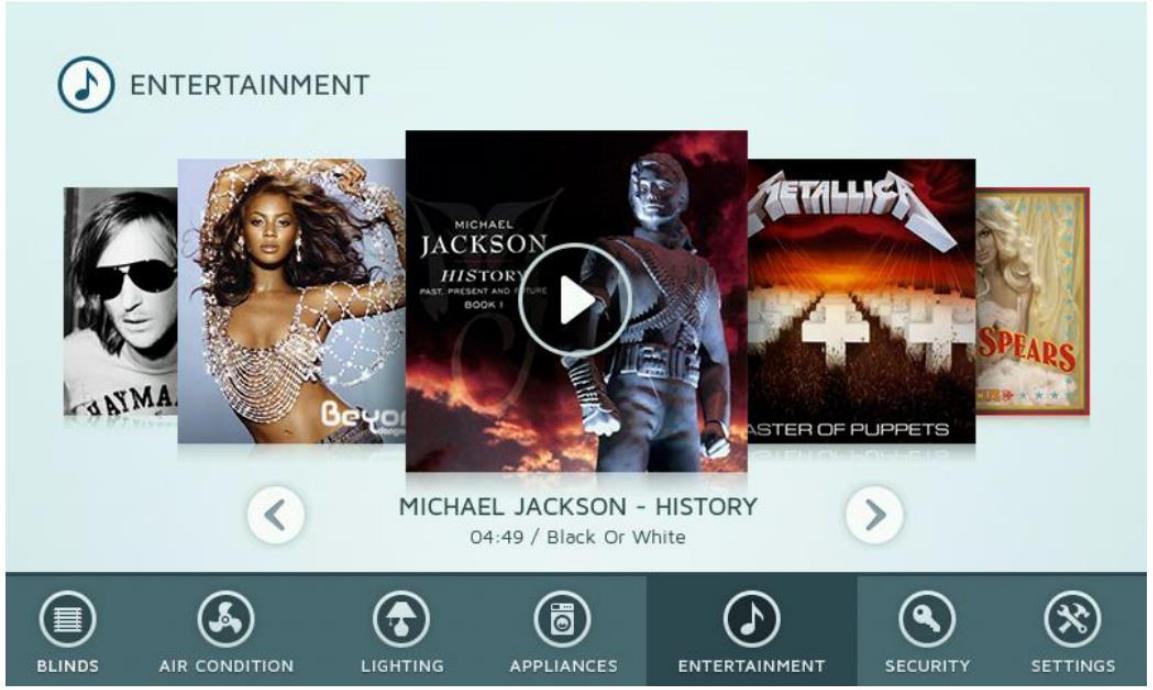
Create custom controls by extending or modifying existing widgets or by combining existing controls with custom functionality.

Advanced Graphical Objects

Draw lines, circles, custom shapes, and graphics, or apply scaling and 3D rotation to images at runtime with highly optimized and memory efficient algorithms.

Requirements

- **Requires modern GUI features like**
  - Transparency
  - Alpha-blending
  - Anti-Aliased Fonts and Kerning
  - Touch Gestures
  - Animations
  - Rotation 2D/3D
  - Screen Transitions
  - High-Resolution Displays
  - High Frame Rate



About TouchGFX

Graphical framework

TouchGFX is a software framework written in C++ that unlocks the graphical user interface on your STM32 hardware.

The technology lets you create high-end GUIs that fully live up to today's smartphone standards at a fraction of the cost.

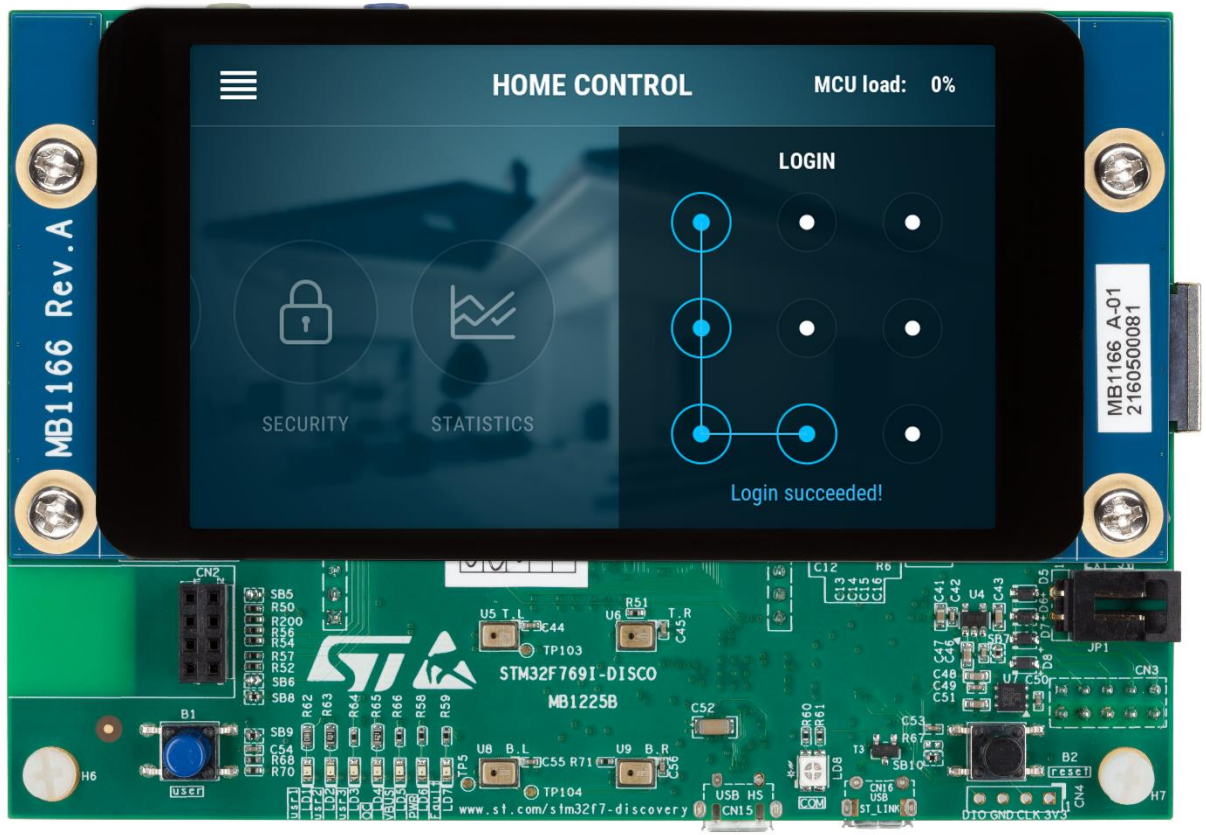
life.augmented

TouchGFX advantages

Create UI with the maximum performance

Create Anything

Through easy creation of custom widgets, TouchGFX is your perfect tool for developing one-of-a-kind applications with a smartphone look and feel.



Max UI Performance on STM32

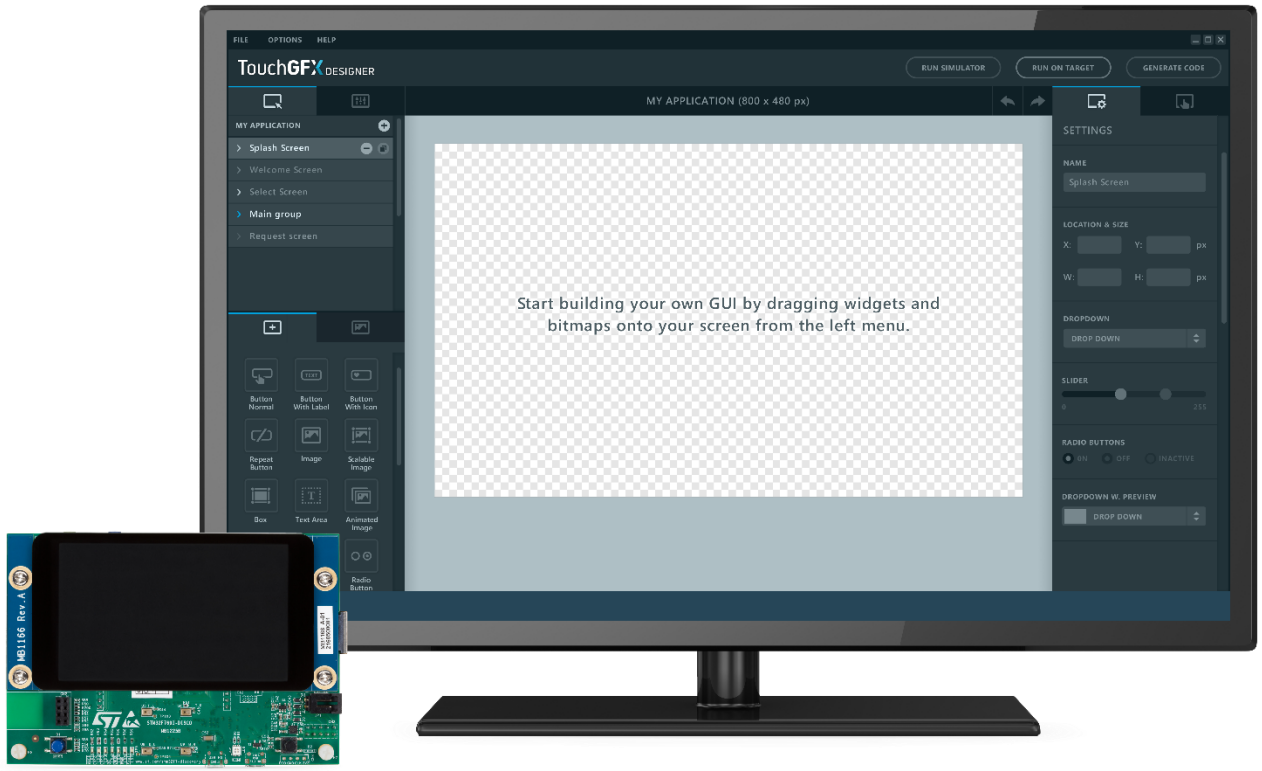
TouchGFX enables astounding smartphone UI performance by full utilization of STM32 advanced graphical hardware features, such as the **Chrom-ART Accelerator™** and **JPEG HW accelerator**.

life.augmented

TouchGFX

TouchGFX designer

State of the art development tool



Easy Development

- Use our graphical WYSIWYG tool, TouchGFX Designer, and create your own prototype in minutes.
- Choose your preferred IDE for development
- Support for all major compilers: IAR, Keil, GCC.
- Run your application on any STM32L4, STM32F4 & F7 display board
- Try before buying: Get our free and fully functional evaluation version.

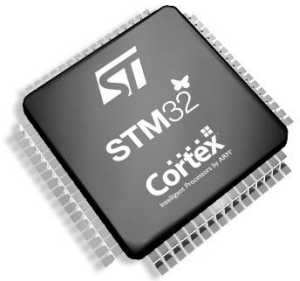
life.augmented

TouchGFX

Application example

Sodastream MIX

- **But most often also requires either**
  - Low cost
  - Low power consumption
  - Low hardware complexity
- **Solution**
  - Microcontroller based design
  - Single chip solution
  - High performance GUI library

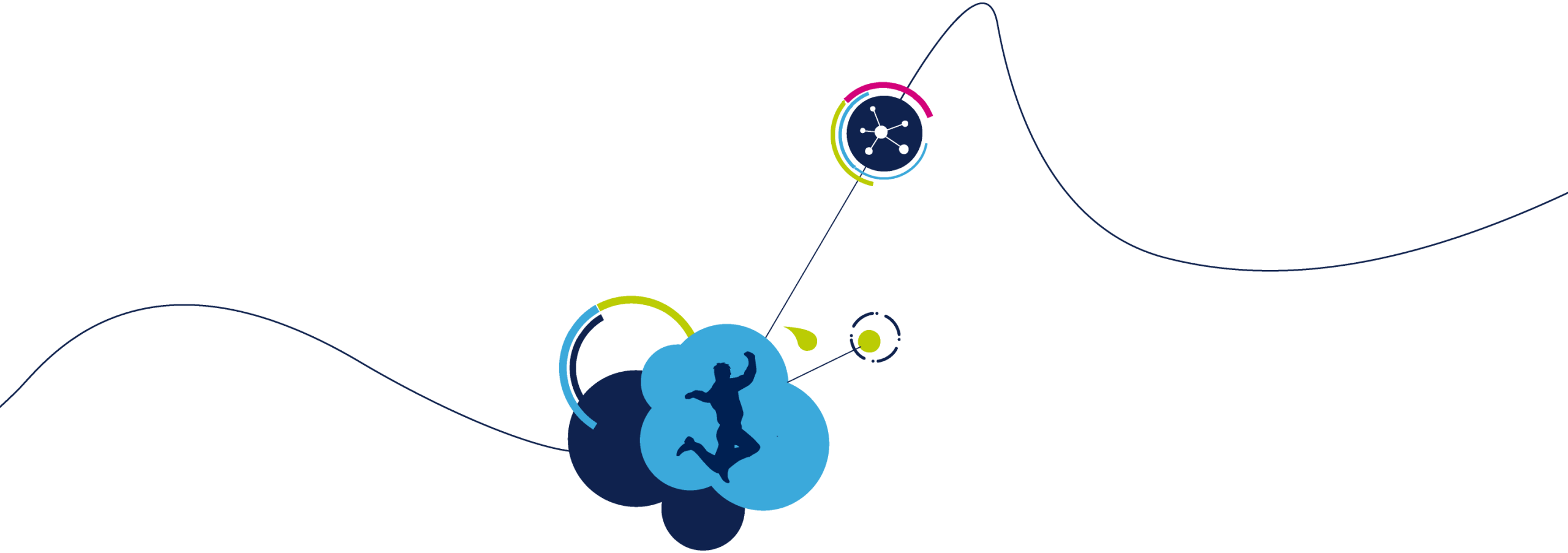


+

TouchGFX

life.augmented

TouchGFX



STM32 & TouchGFX

life.augmented

TouchGFX

STM32 GFX acceleration technology

Key acceleration IPs

- **Chrom-ART Accelerator™**
  TouchGFX utilizes Chrom-ART to the max and offloads the CPU from repetitive graphic tasks.
  - Efficient 2D Image copy
  - Transparency
  - Picture format conversion

STM32 interfacing technology

Key interfaces for displays and memories

- **RGB TFT interface**
  TouchGFX supports the STM32 LTDC to use any classical RGB displays.

- **DSI Interface**
  TouchGFX supports the STM32 DSI peripheral to enable the use of DSI displays.

- **SDRAM 16 and 32 bits**
  32 bits provide higher bandwidth giving better UI performance.

- **Memory-mapped Quad SPI Flash**
  Enables very fast rendering as TouchGFX can render directly from the Quad SPI flash into the framebuffer.
  Reduces BOM, pin count and board complexity compared to parallel NOR.

STM32 tooling technology

Key tools integration

- **Cube**
  TouchGFX runs on top of the cube driver layer. Use CubeMX to generate low-level code for your custom board.

STM32 step 1 portfolio

TouchGFX supported devices

STM32 discovery kit

TouchGFX supported ST Discovery Kits

STM32F429IDISCOVERY

- STM32F429
- 320x240 QVGA LCD

STM32F469IDISCOVERY

- STM32F469
- 800x480 WVGA LCD

STM32F746GDISCOVERY

- STM32F746
- 480x272 WQVGA LCD

STM32F769IDISCOVERY

- STM32F769
- 800x480 WVGA LCD

life.augmented

TouchGFX

STM32 evaluation boards

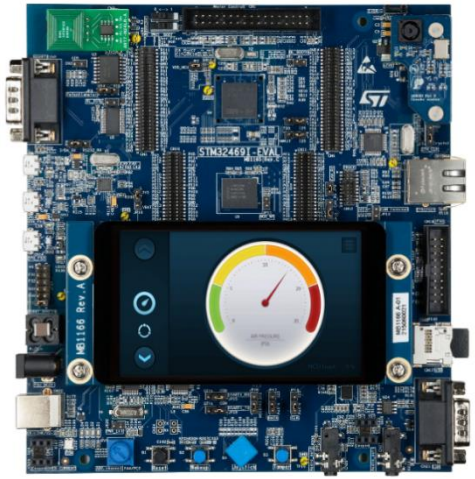
TouchGFX supported ST Eval Boards

STM32429I-EVAL

- STM32F429
- 480x272 WQVGA LCD
- 256 MB SDRAM
- 128 MB NOR Flash

STM32439I-EVAL

- STM32F439
- 640x480 VGA LCD
- 256 MB SDRAM
- 128 MB NOR Flash



STM32469I-EVAL

- STM32F469
- 800x480 WVGA LCD
- 256 MB SDRAM
- 128 MB NOR Flash
- 512 MB QSPI Flash

STM32756G-EVAL

- STM32F756
- 640x480 VGA LCD
- 256 MB SDRAM
- 128 MB NOR Flash
- 512 MB QSPI Flash

STM32769G-EVAL

- STM32F769
- 800x480 WVGA LCD
- 256 MB SDRAM
- 128 MB NOR Flash
- 512 MB QSPI Flash

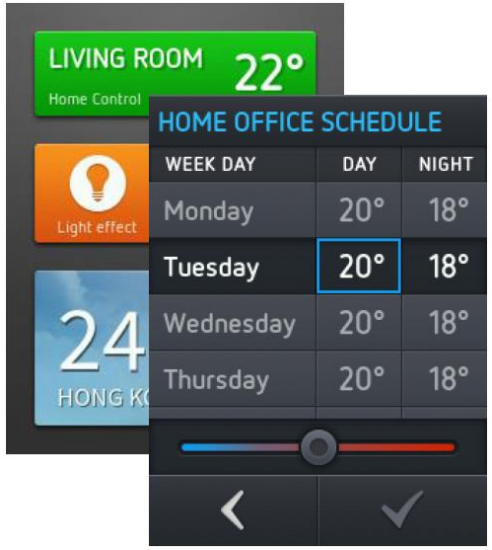
life.augmented

TouchGFX

STM32 demonstrations

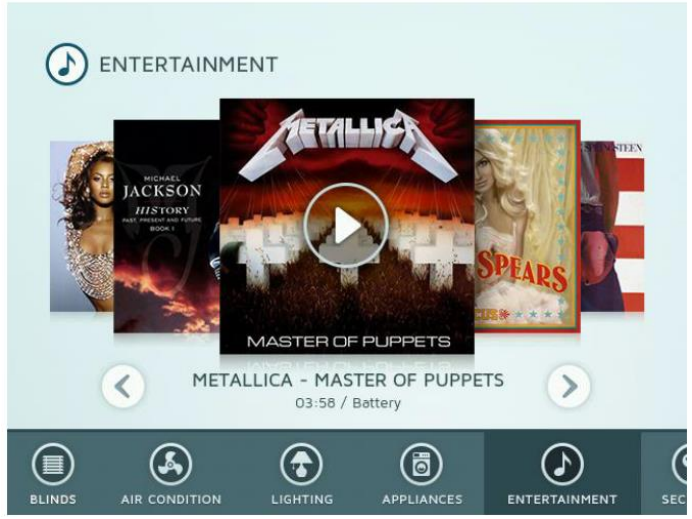
All demos are available for download in binary format

Demo - 2.4” (STM32F429-DISCOVERY)



Demo - 4.3” (STM32F429I-EVAL)

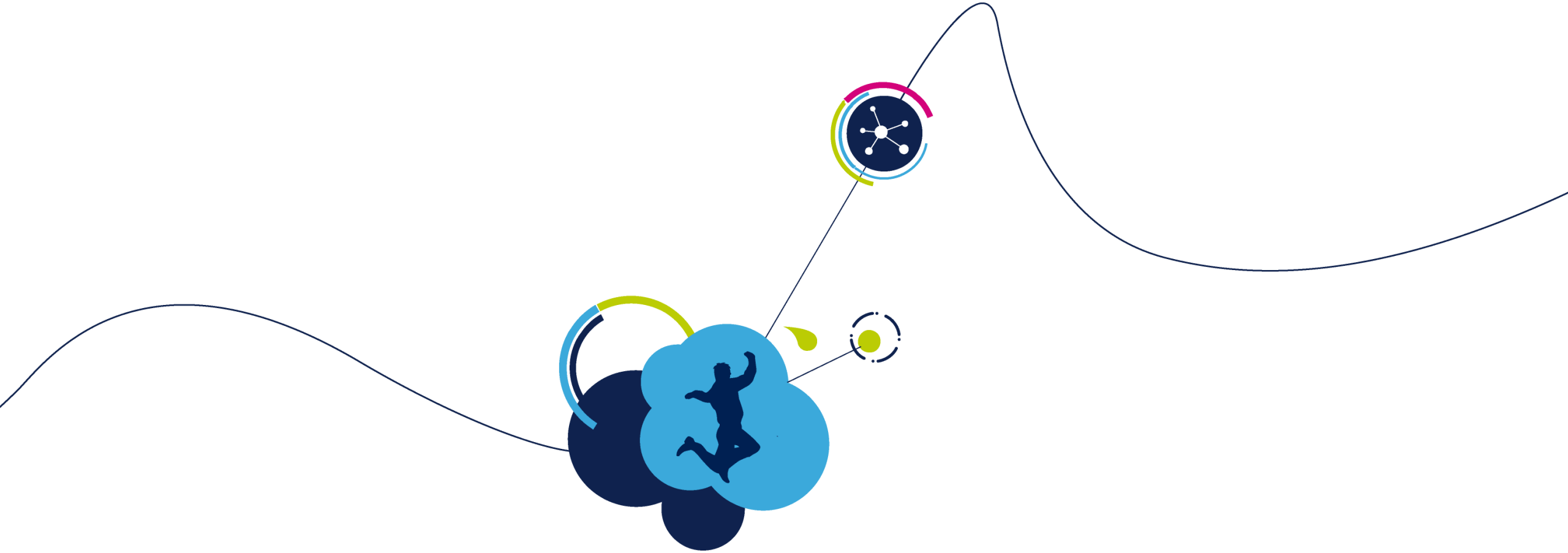
Demo - 5.7” (STM32F4x9I-EVAL)



See http://touchgfx.com/product-details/demos/

life.augmented

TouchGFX



TouchGFX Development

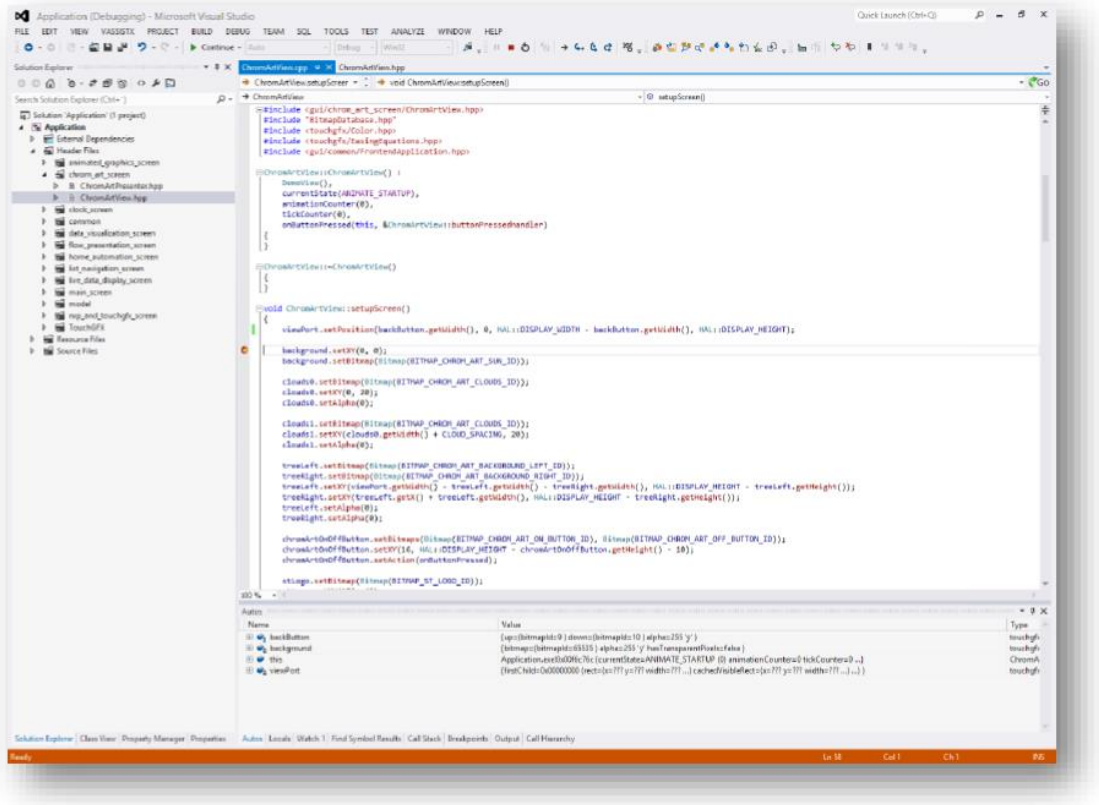
life.augmented

TouchGFX

Development process

TouchGFX environment

- **C++ framework**
  - TouchGFX Core
  - Board-specific hardware abstraction layer
  - No external libraries required
- **Compilers**
  - Simulator –MS Visual Studio, GCC (Make)
  - Target –IAR, Keil, GCC
- **Development Environment**
  - Simulator –MS Visual Studio (Eclipse, others)
  - Target –IAR Workbench, Keil µVision



TouchGFX development

TouchGFX tools

TouchGFX combines the simplicity of a WYSIWYG designer, the efficiency and flexibility of the C++ language with the convenience of a PC simulator to create the perfect environment for developing advanced and rich graphical interfaces, fast.

- **Text Converter**
  Manage multiple languages and converts texts and translations into an optimized target format
- **Image Converter**
  Converts application images to target format optimized for Chrom-ART accelerator™
- **Font Converter**
  Converts .ttf font files into target format optimized for the Chrom-ART accelerator™
- **TouchGFX Designer**
  WYSIWYG graphical drag and drop tool
- **PC Simulator**
  For easy validation and testing of your application during development

Development process

The TouchGFX distribution - tools

- **Font Converter**
  - TrueType Fonts
- **Image Converter**
  - PNG, BMP
  - Alpha per pixel support
- **Text database / Multiple language support**

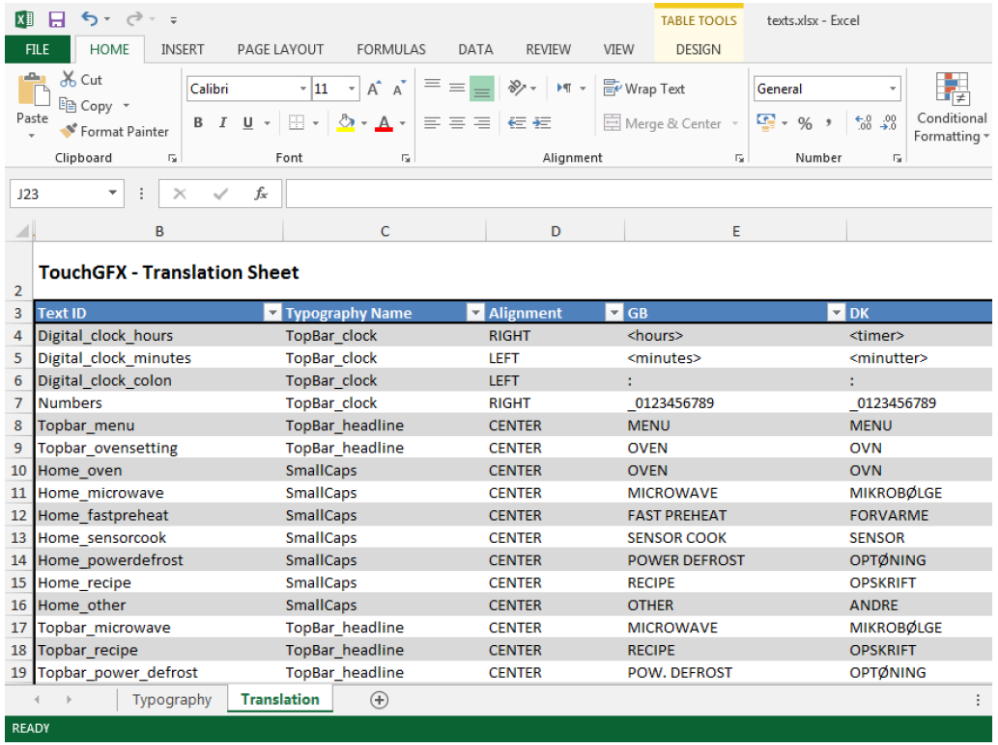


TABLE TOOLS texts.xlsx - Excel

FILE HOME INSERT PAGE LAYOUT FORMULAS DATA REVIEW VIEW DESIGN

Cut Copy Format Painter Paste — Clipboard

Calibri 11 — Font

Wrap Text Merge & Center — Alignment

General — Number

Conditional Formatting

J23

TouchGFX - Translation Sheet

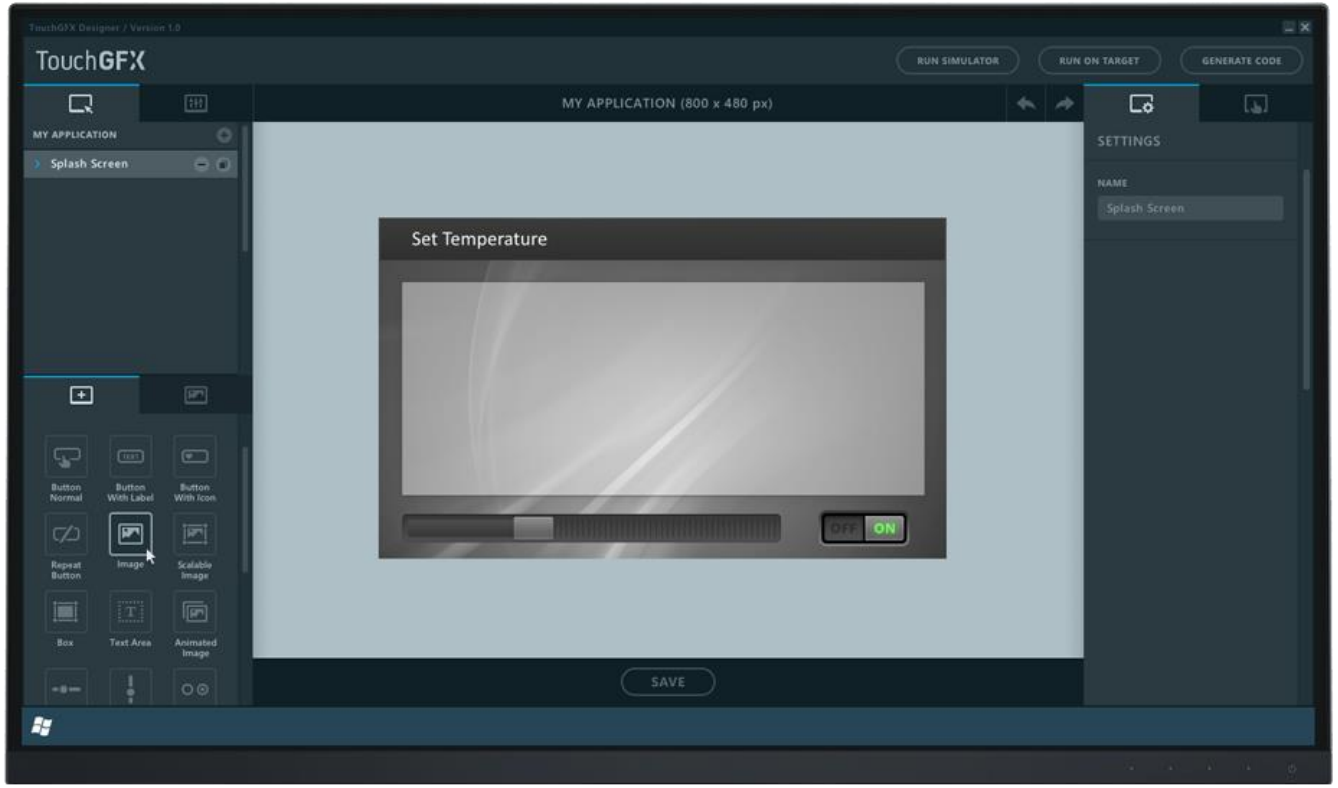
Text ID	Typography Name	Alignment	GB	DK
Digital_clock_hours	TopBar_clock	RIGHT	<hours>	<timer>
Digital_clock_minutes	TopBar_clock	LEFT	<minutes>	<minutter>
Digital_clock_colon	TopBar_clock	LEFT	:	:
Numbers	TopBar_clock	RIGHT	_0123456789	_0123456789
Topbar_menu	TopBar_headline	CENTER	MENU	MENU
Topbar_ovensetting	TopBar_headline	CENTER	OVEN	OVN
Home_oven	SmallCaps	CENTER	OVEN	OVN
Home_microwave	SmallCaps	CENTER	MICROWAVE	MIKROBØLGE
Home_fastpreheat	SmallCaps	CENTER	FAST PREHEAT	FORVARME
Home_sensorcook	SmallCaps	CENTER	SENSOR COOK	SENSOR
Home_powerdefrost	SmallCaps	CENTER	POWER DEFROST	OPTØNING
Home_recipe	SmallCaps	CENTER	RECIPE	OPSKRIFT
Home_other	SmallCaps	CENTER	OTHER	ANDRE
Topbar_microwave	TopBar_headline	CENTER	MICROWAVE	MIKROBØLGE
Topbar_recipe	TopBar_headline	CENTER	RECIPE	OPSKRIFT
Topbar_power_defrost	TopBar_headline	CENTER	POW. DEFROST	OPTØNING

Typography Translation

READY

TouchGFX designer

From prototype to product



Creation of screen in TouchGFX Designer

TouchGFX Designer will support you throughout your entire UI project by simplifying the process of creating the visual design and layout of your screens and custom controls.

Your TouchGFX application code is automatically updated with the changes done in the Designer.

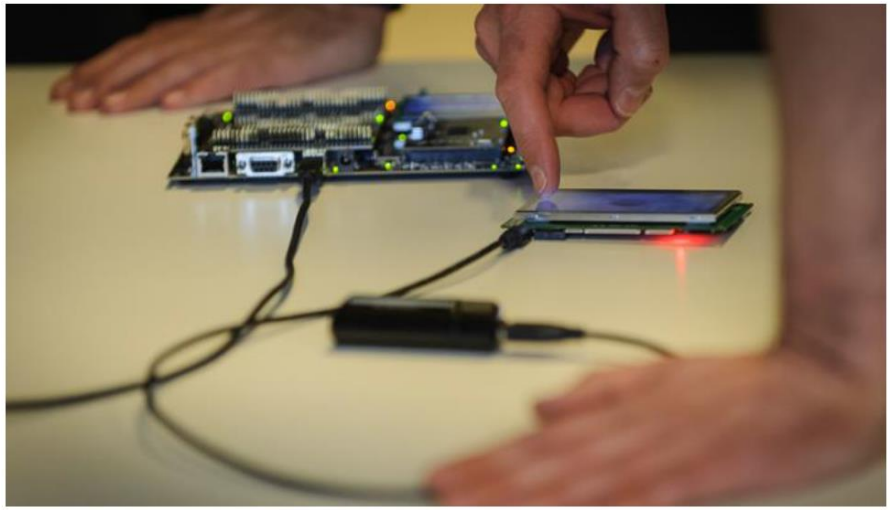
life.augmented

TouchGFX

Development process

The TouchGFX distribution – build, download and debug

- **PC simulator (Windows, Linux)**
  - Runs the same code as target
  - Main development platform for the GUI
  - Debug on PC
  - Test
  - Validation
  - Training



- **Target support**
  - GCC (free), IAR (license) and Keil (license) support
  - Evaluation board support –compile, download and run ”off the shelf”
  - Easy download through IAR or other external tools
  - Debug on target integrated in IAR Workbench, Keil µVision and GCC (GDB)

Development process

The TouchGFX distribution – Demo and examples

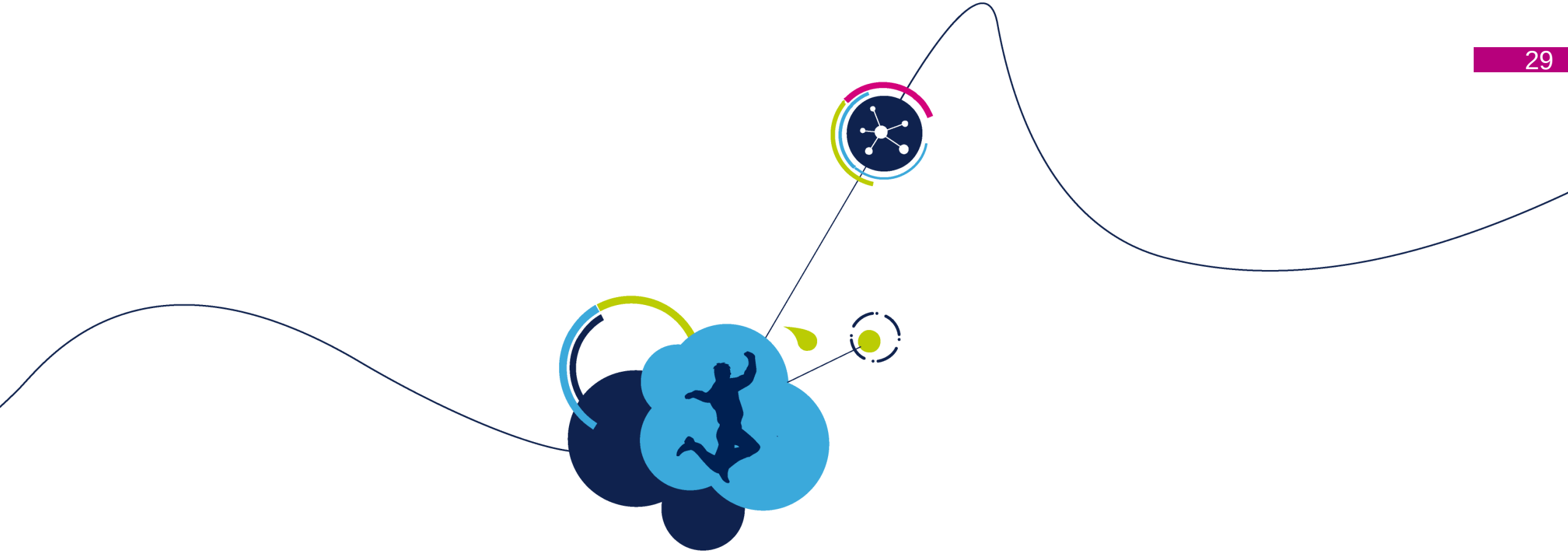
- **Examples:**
  - A lot of small easy-to-understand examples
  - Show basic features and widgets
  - Full source code included in the TouchGFX eval version
- **Demos:**
  - More advanced applications than the examples
  - Gives a good impression of how to do complex graphics
  - Full source code included in the TouchGFX eval version

Development process

The TouchGFX distribution – Documentation

- **Help Center**
  - Getting started
  - Knowledge Base
  - Help Desk
  - Community
  - Manual

- **http://support.touchgfx.com**



TouchGFX Technical Overview

Hardware setup

Required Hardware

- **Low MCU Load**
  - Typical< 20%
  - The MCU can perform other application tasks.
  - Single-chip solution.
- **Low Memory Footprint**
  - **Internal RAM**: 10-35 kB (framework, stack, widgets).
  - **Internal flash**: 20 kB (framework) + 1 –80 kB (screen definitions, UI logic).
  - **(Internal or External) RAM**: 150 kB -2 MB (one or two frame buffers depending on the display).
  - **(External) Flash**: 1-8 MB (graphics data, fonts, text strings). May be more….
- **Can be used on any STM32**
  - Targeting STM32 with TFT controller.
- **Display Resolutions:**
  - **QVGA**: 320x240 (often3.5”).
  - **WQVGA**: 480x272 (often4.3”).
  - **WVGA**: 800x480 (often7.0”).
  - **WSVGA**: 1024x600 (often10”).

Targeted platform

With or without OS, any display type

- **Operating system**

  OS independent:

  - Runs on any RTOS (uses just one task and two semaphores)
  - Runs on Bare Metal (no OS)

- **Display types**

  Support for various display types, e.g.:

  - Parallel RGB
  - MIPI-DSI
  - 8080/SPI

Framework focus : MCU load

Two approaches to representing graphics

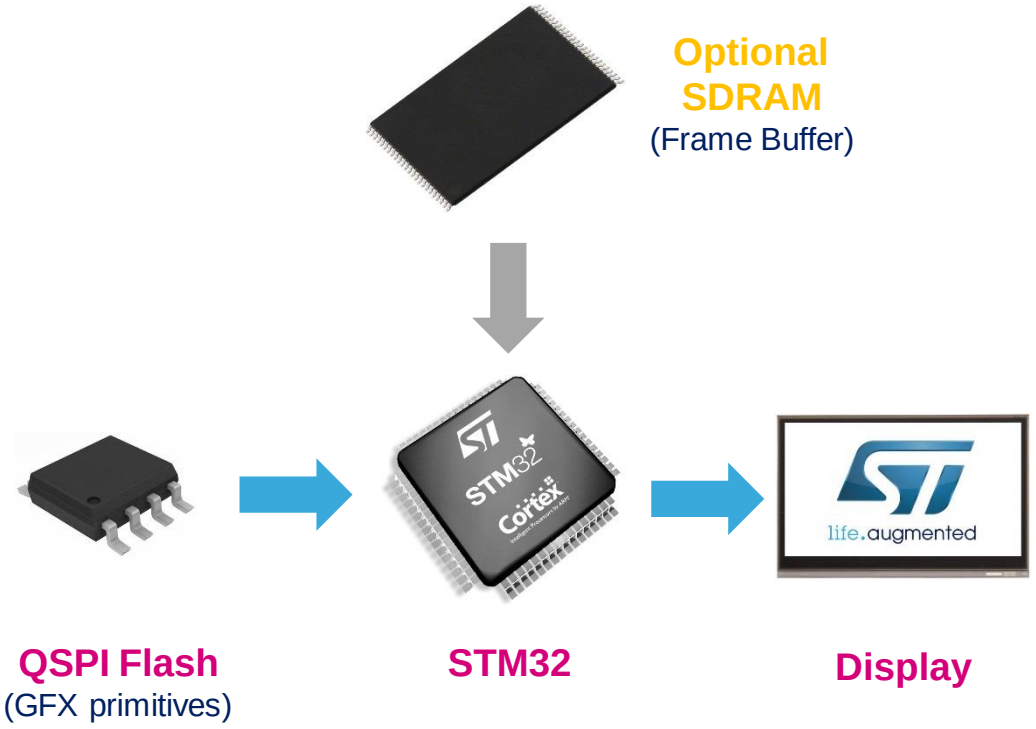
- **Bitmap/raster graphics**
  - Pre-rendered.
  - At runtime basically just a data copy
- **Much faster and can make heavy use of ChromART transfers**

- **Arithmetic/Vector graphics**
  - Run-time calculations and pixel plotting by MCU
  - More dynamic
- **Not really an option in this microcontroller context**

The TouchGFX framework

Framework focus : MCU load

- **Key decision: Based on bitmap graphics**
  - Heavy use of Chrom-ART accelerator
  - Fast visible surface determination
  - No standard graphics
- **But the goal is to create good-looking dynamic UIs!**
  - The framework helps the user achieve this
  - Full control of the widgets and the redrawing mechanism
  - Custom widgets and containers
  - Dynamic content equals many bitmaps
  - But flash is cheap
  - Also offer direct pixel manipulation



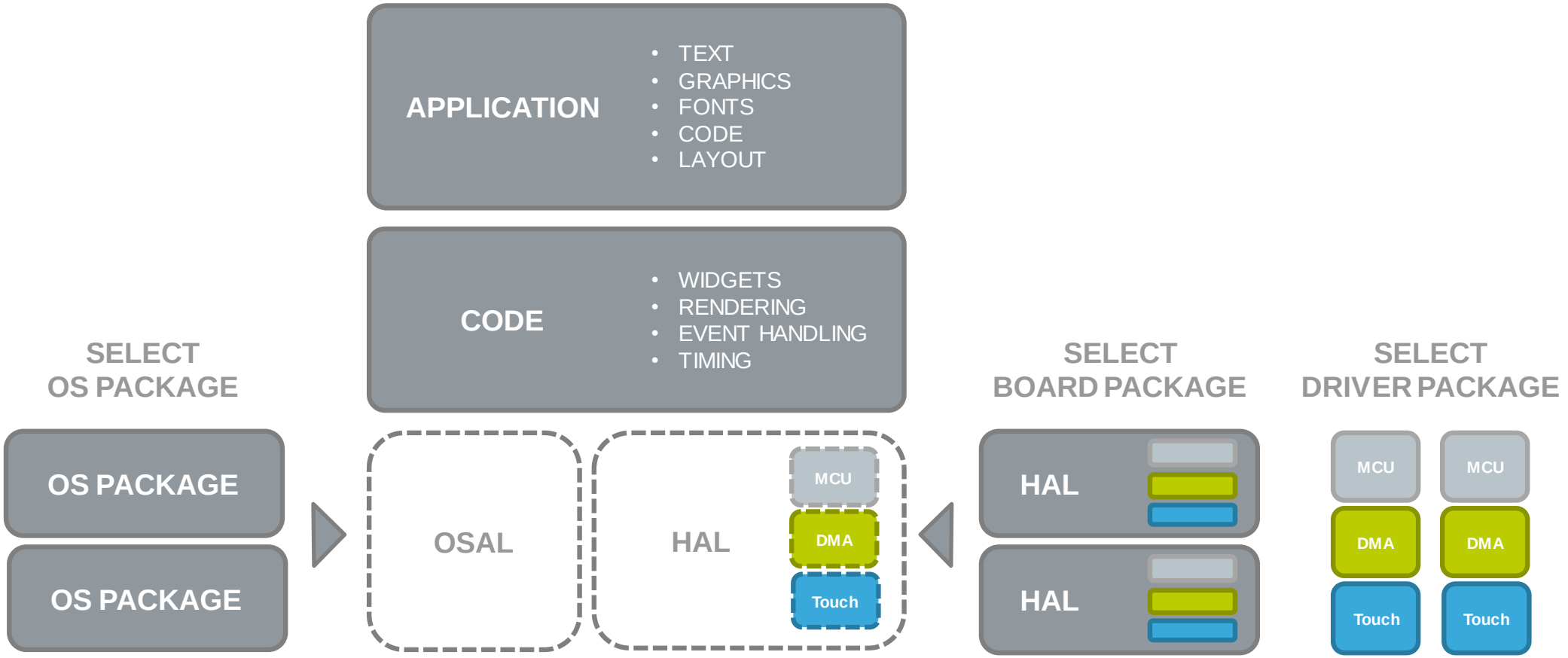
The TouchGFX framework

Framework focus : Low memory footprint

- **No dynamic memory allocation**
  - Real time concerns
  - Predictability
  - Certification
- **Memory usage of the largest screen (Presenter/View pair) is determined at compile time**
- **Memory usage:**
  - Internal RAM: 10-35 kB (framework, stack, widgets).
  - (Internal) flash: 20 kB (framework) + 1 –80 kB (screen definitions, UI logic).
  - (Internal or External) RAM:150 kB -2 MB (one or two frame buffers depending on the display architecture).
  - (External) Flash: 1-8 MB (graphics data, fonts, text strings).

The TouchGFX framework

Software layers

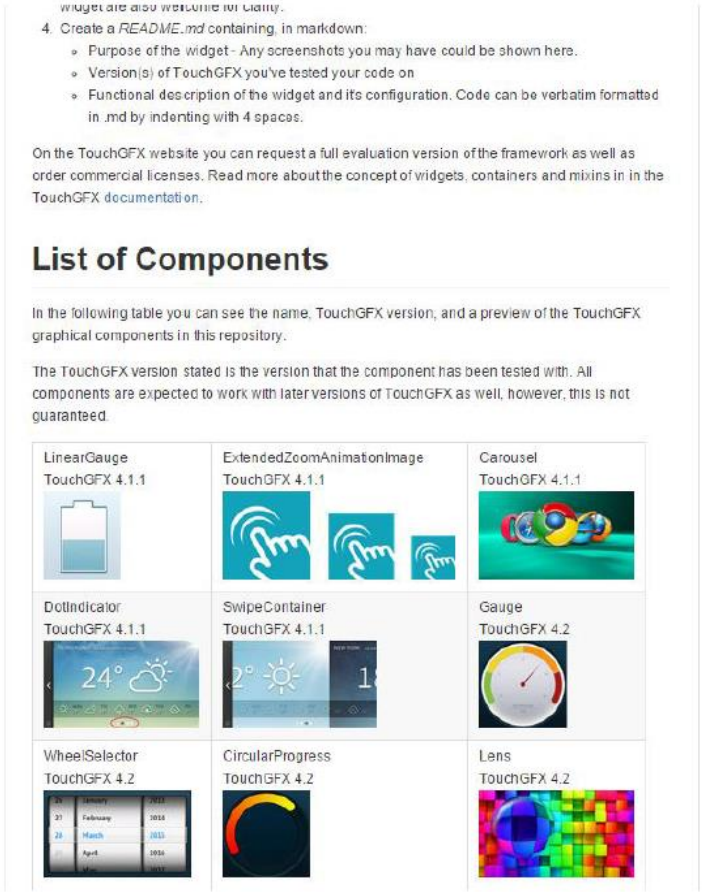


Development process

TouchGFX open repository

- **GitHub repository**
  - Contains numerous examples of, and ideas for, graphical components such as widgets, containers and mixins.
  - Code can be used directly or as inspiration for your applications
  - Free of charge

- **https://github.com/draupnergraphics/touchgfx-open-repository**


widget are also welcome for clarity.

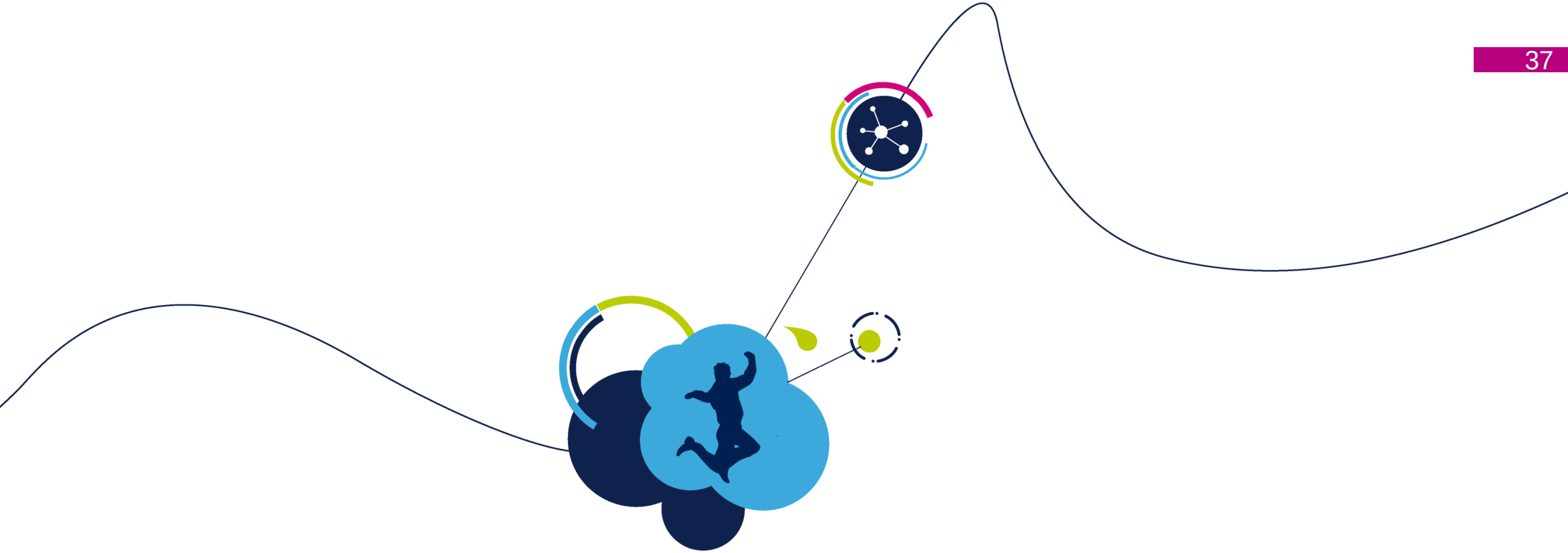
4. Create a *README.md* containing, in markdown:
   - Purpose of the widget - Any screenshots you may have could be shown here.
   - Version(s) of TouchGFX you've tested your code on
   - Functional description of the widget and its configuration. Code can be verbatim formattad in .md by indenting with 4 spaces.

On the TouchGFX website you can request a full evaluation version of the framework as well as order commercial licenses. Read more about the concept of widgets, containers and mixins in in the TouchGFX documentation.

List of Components

In the following table you can see the name, TouchGFX version, and a preview of the TouchGFX graphical components in this repository.

The TouchGFX version stated is the version that the component has been tested with. All components are expected to work with later versions of TouchGFX as well, however, this is not guaranteed.



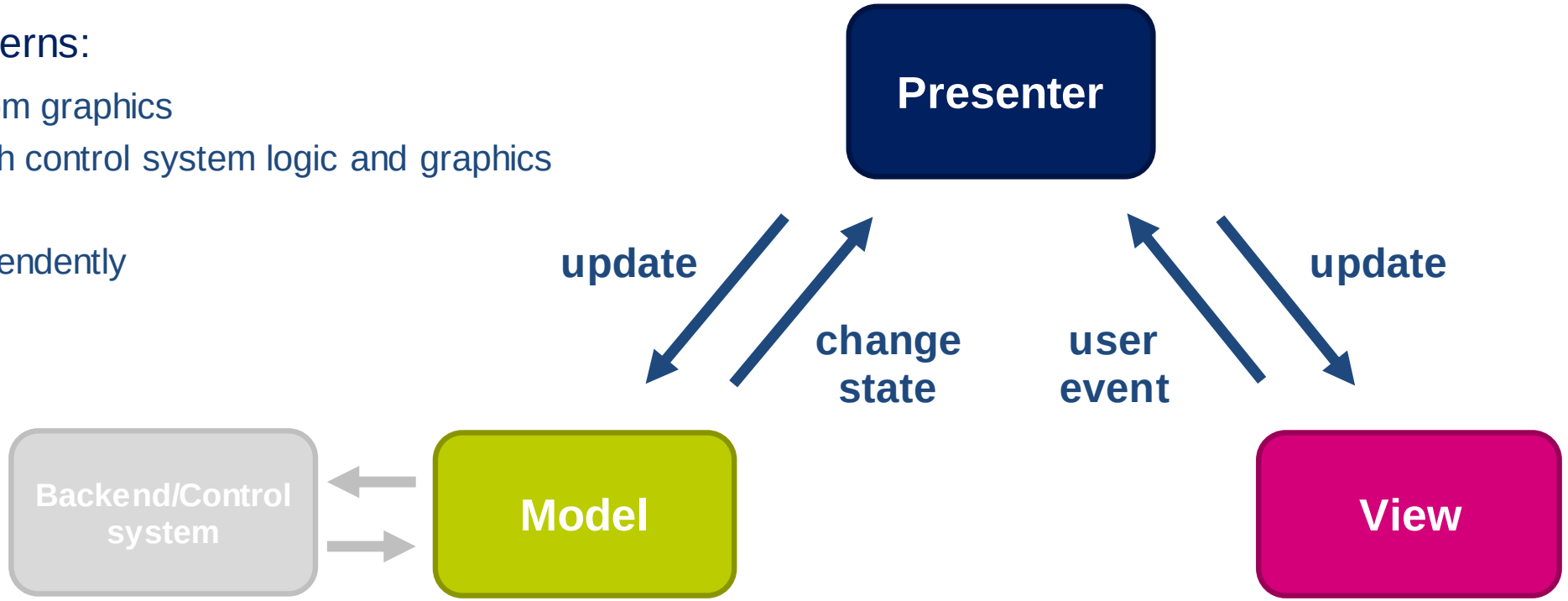
TouchGFX Framework

MVP Design Pattern

Model – View – Presenter

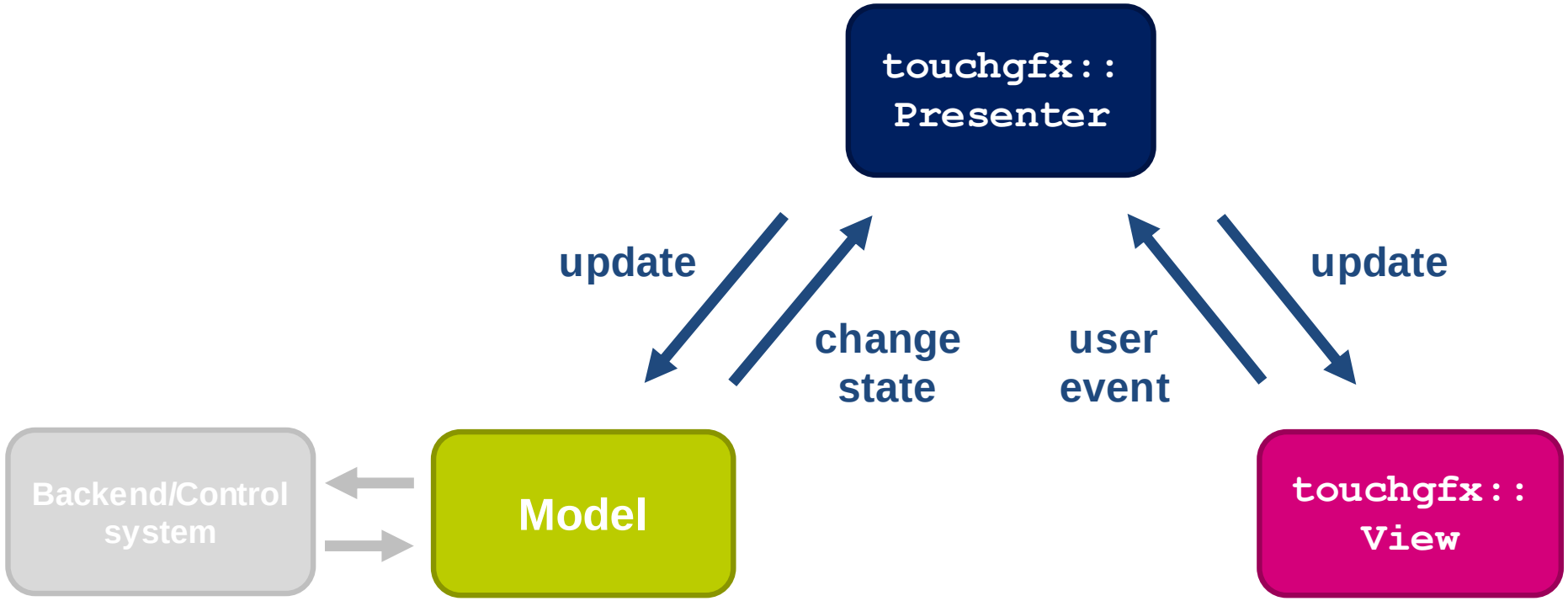
Model-View-Presenter Software Architecture

- Well known design pattern
- Separation of concerns:
  - Logic separated from graphics
  - Communication with control system logic and graphics
  - Reuse of code
  - Easier to test independently



TouchGFX MVP Classes

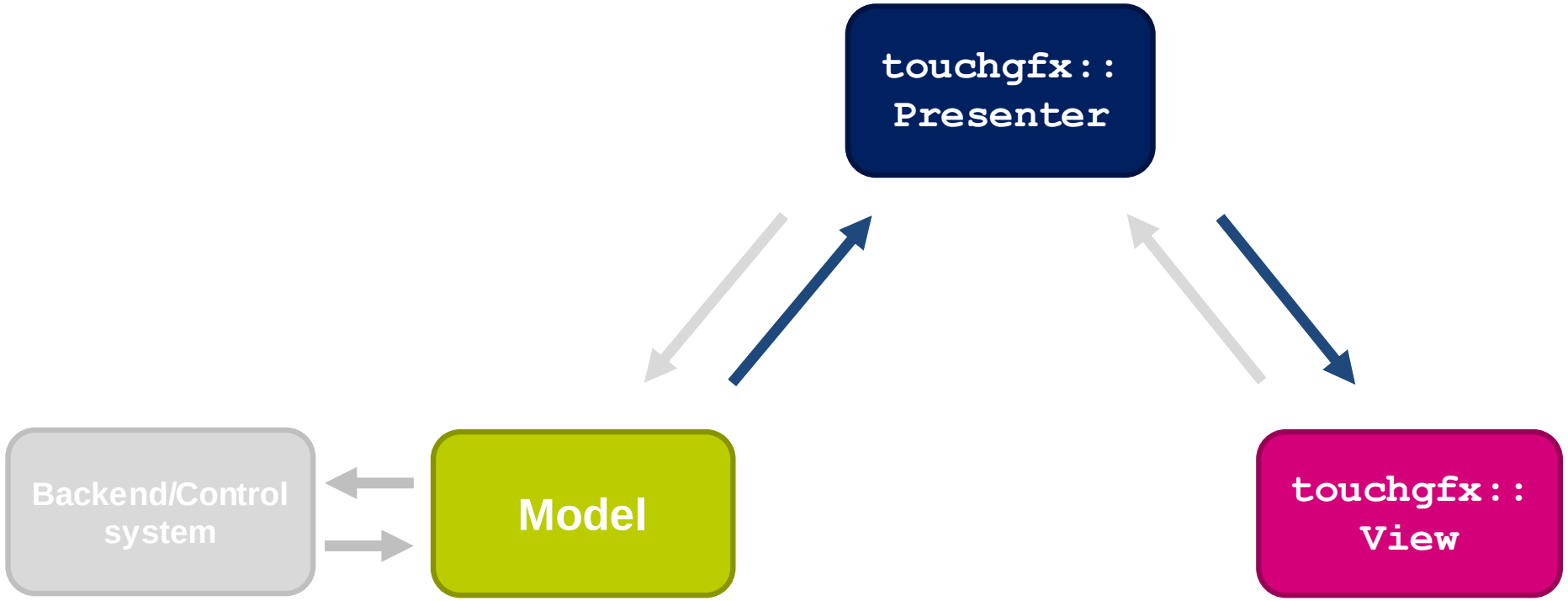
Model – View – Presenter



MVP Initial Code

Model – View – Presenter

```
void TemperaturePresenter::activate()
{
    view.setMeasuredTemperature(model->getMeasuredTemperature());
}
```



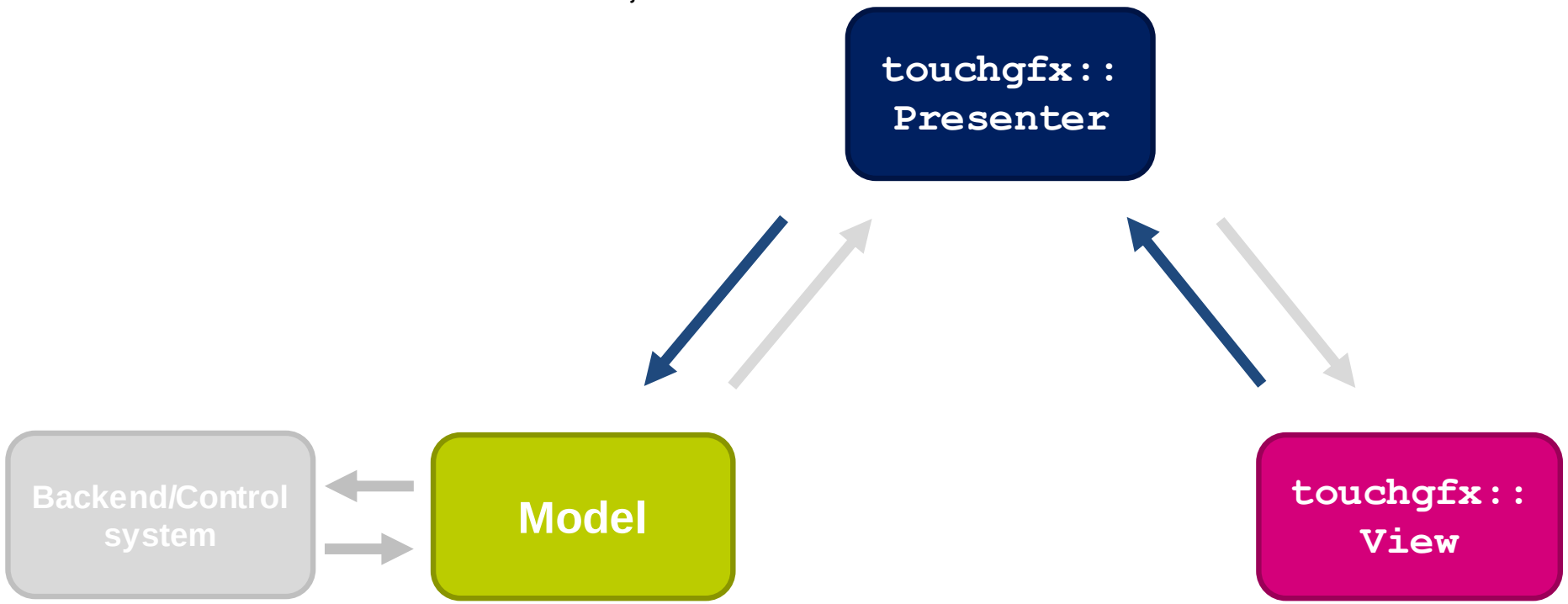

```
int Model::getMeasuredTemperature()
{
    return measuredTemperature;
}
```

```
void TemperatureView::setMeasuredTemperature(int temperature)
{
    temperatureWidget.setMeasuredTemperature(temperature);
}
```

MVP User Action Code

Model – View – Presenter

```
void TemperaturePresenter::adjustDesiredTemperature(bool up)
{
    model->adjustDesiredTemperature(up);
}
```




```
int Model::adjustDesiredTemperature(bool up)
{
    // communicate with external system
}
```

```
void TemperatureView::setMeasuredTemperature(cosnt touchgfx::AbstractButton& btn)
{
    presenter->adjustDesiredTemperature(&btn == &upButton);
}
```

Thanks!

For more information please kindly visit

www.st.com/stm32

www.stmcu.com.cn

www.stmcu.org

life.augmented