A MODERN DATAFLEX LIBRARY STRATEGY

JOHAN BRODDFELT

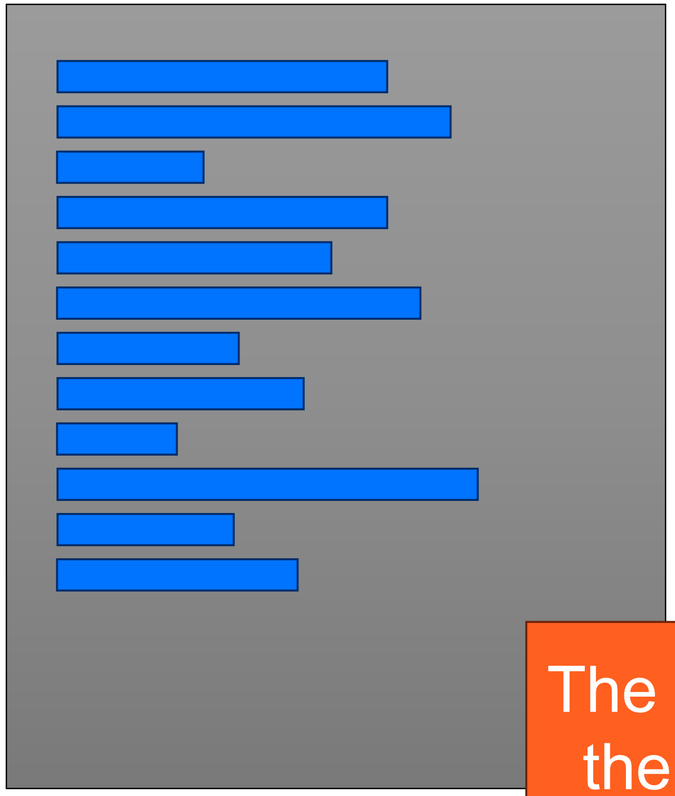
CHAPTERS
OVERVIEW

- Is a library hiding in your code?
- Creating a library
- Manage dependencies
- Community libraries
- Version management
- Good library practices

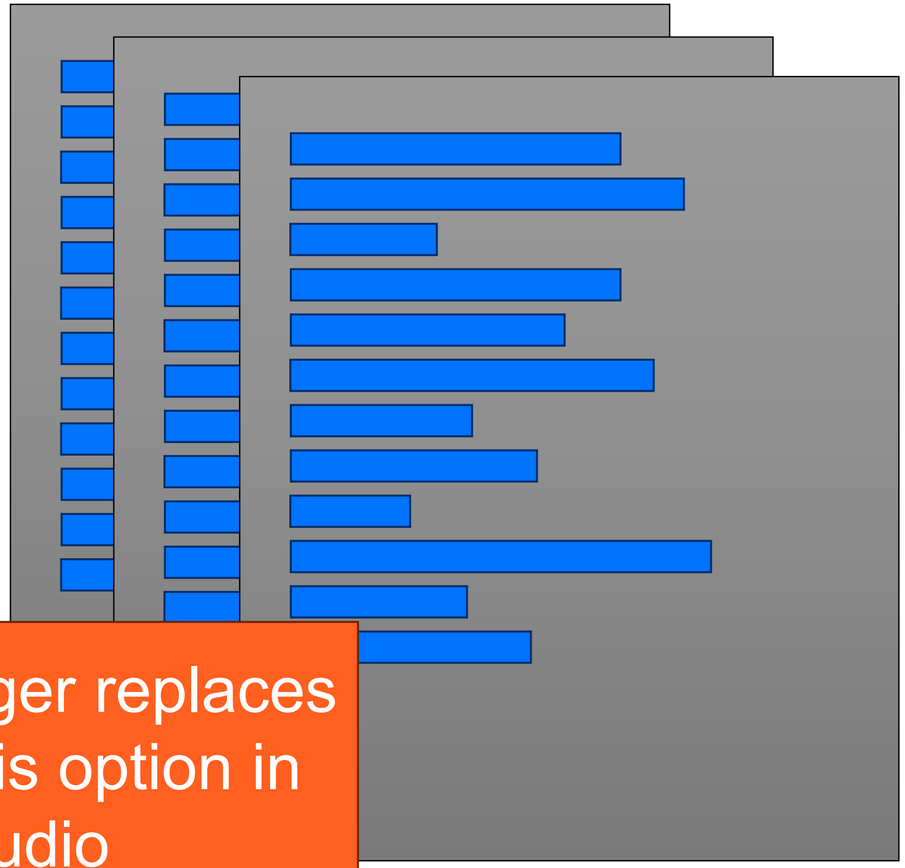
IDENTIFYING A LIBRARY

IDENTIFYING A LIBRARY
PACKAGE OR LIBRARY?

MyPackage.pkg



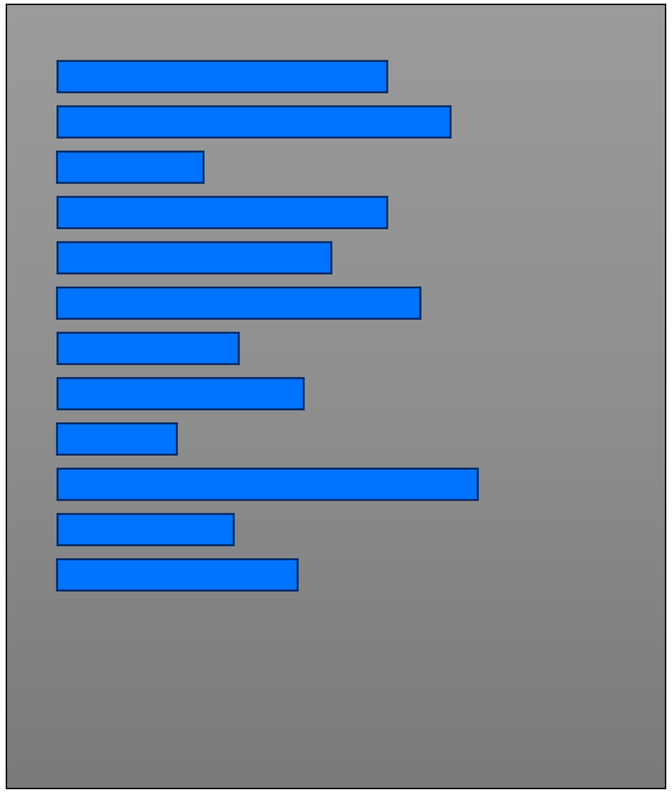
MyLibrary.sws



The Package manager replaces the Maintain Libraris option in DataFlex Studio

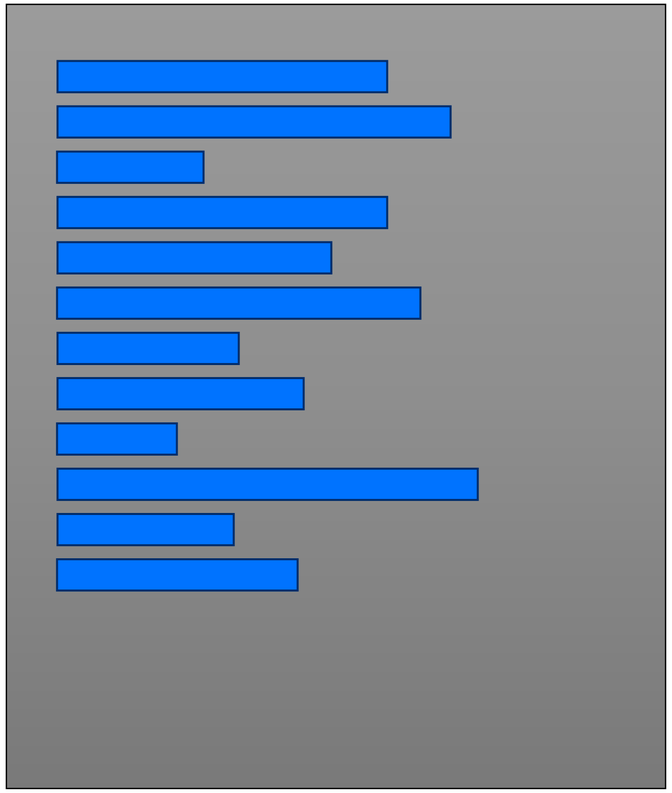
IDENTIFYING A LIBRARY
YOUR CODE

Project A

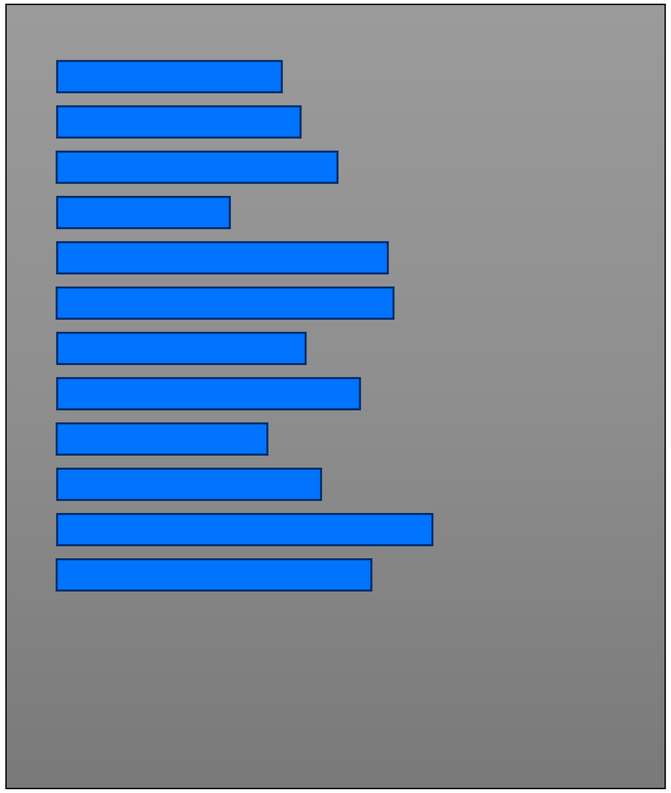


IDENTIFYING A LIBRARY
ADDITIONAL PROJECT

Project A

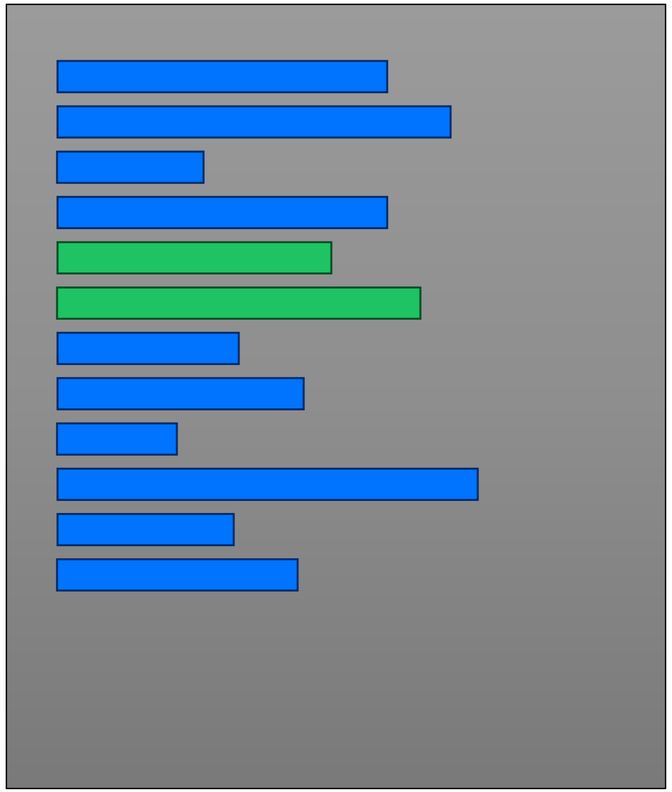


Project B

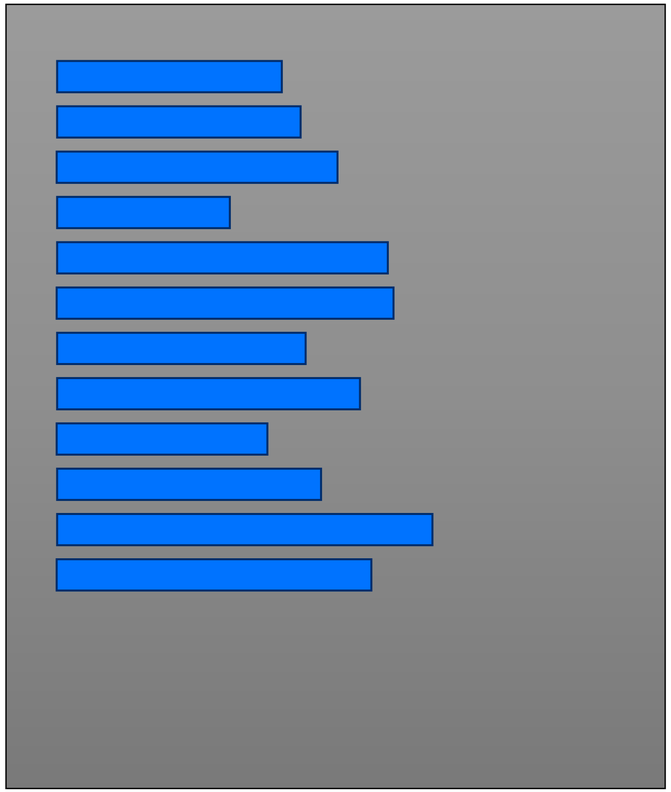


IDENTIFYING A LIBRARY
REUSABLE CODE

Project A

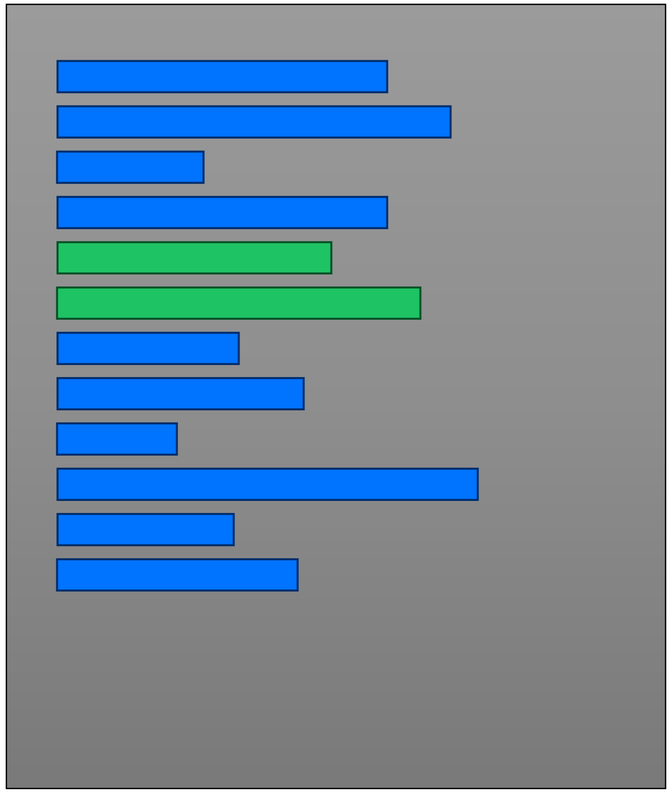


Project B

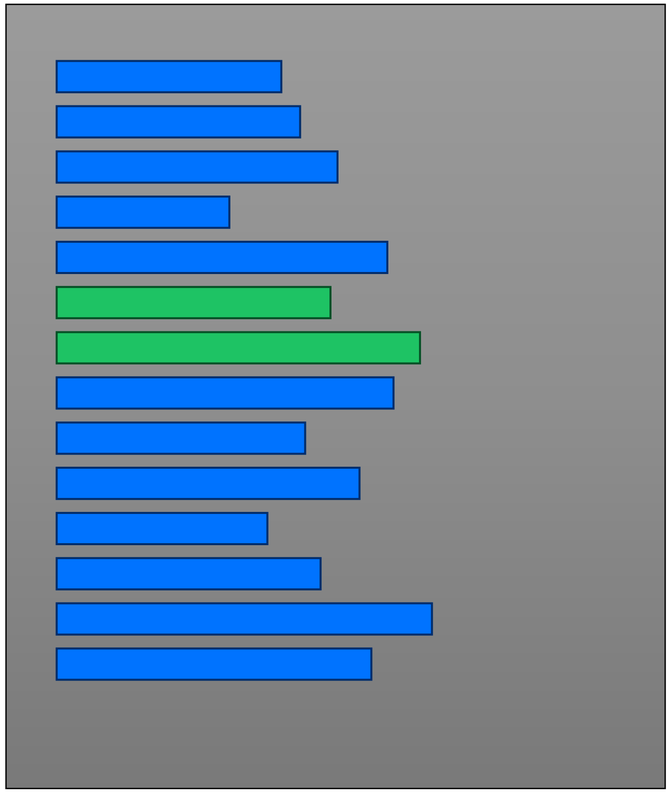


IDENTIFYING A LIBRARY
COPY CODE

Project A

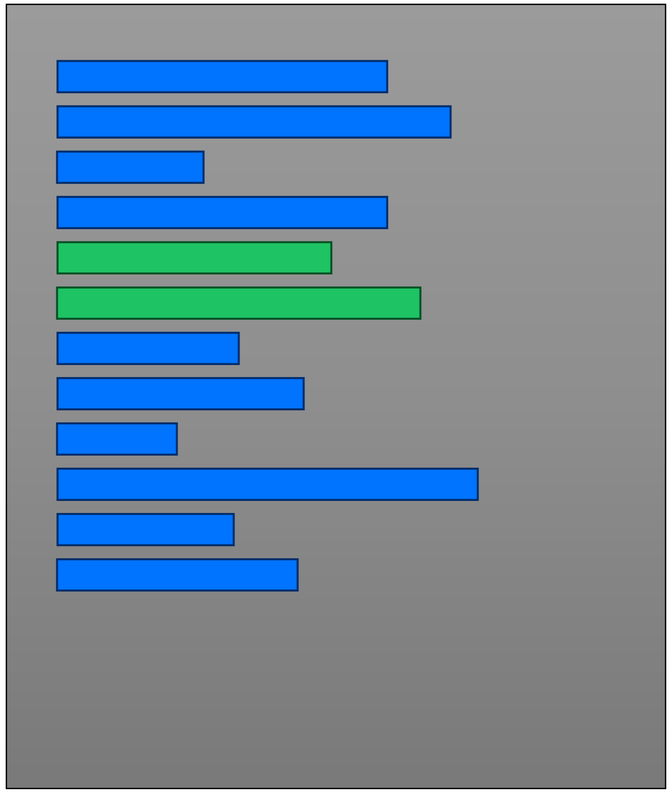


Project B

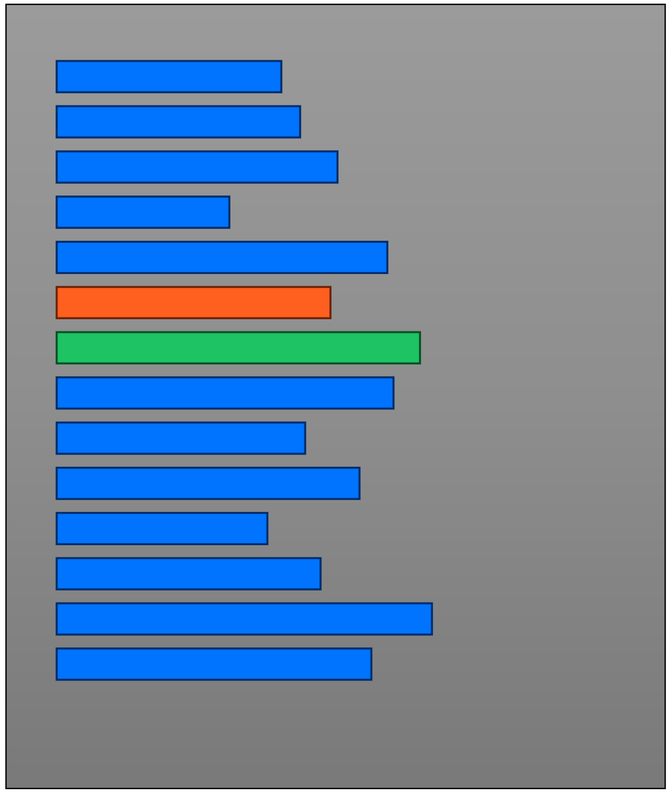


IDENTIFYING A LIBRARY
MODIFIED COPY

Project A

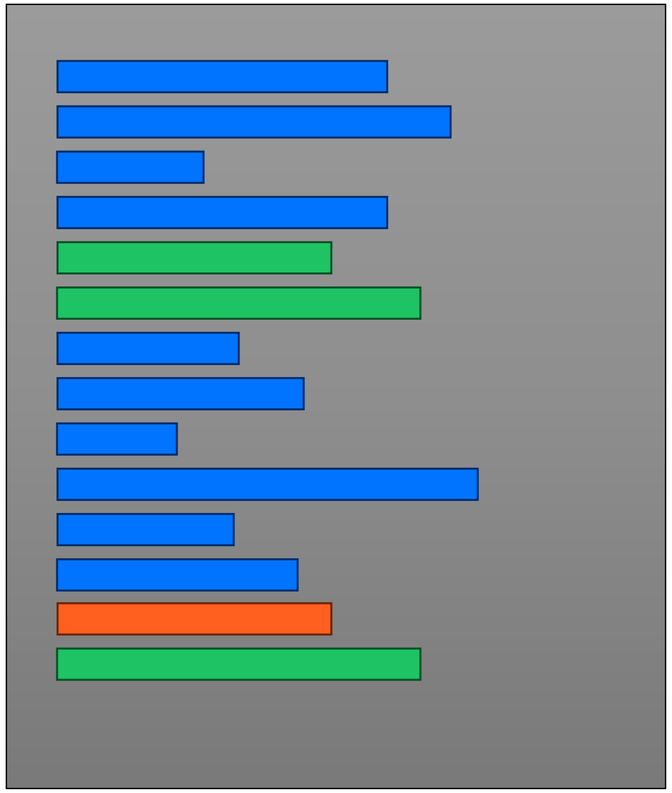


Project B

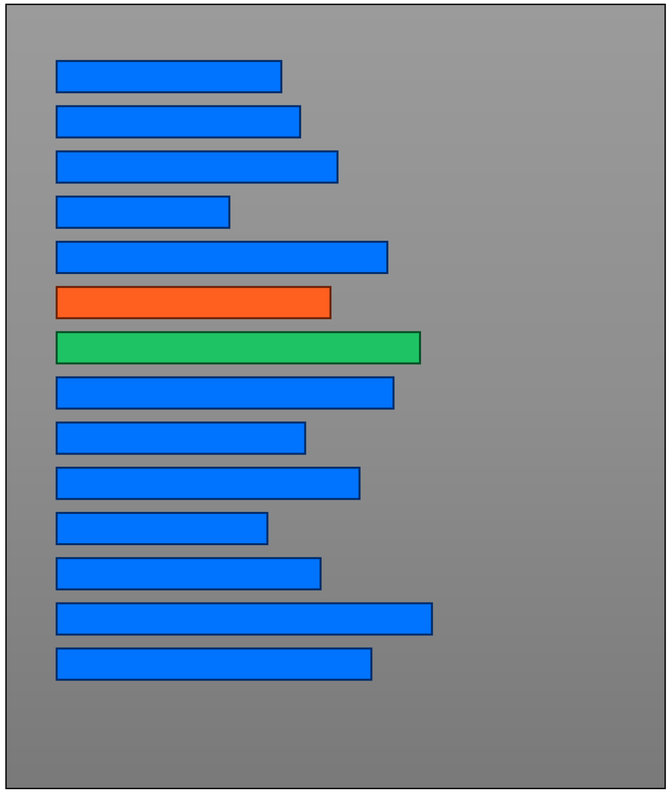


IDENTIFYING A LIBRARY
USE MODIFIED COPY IN FIRST PROJECT

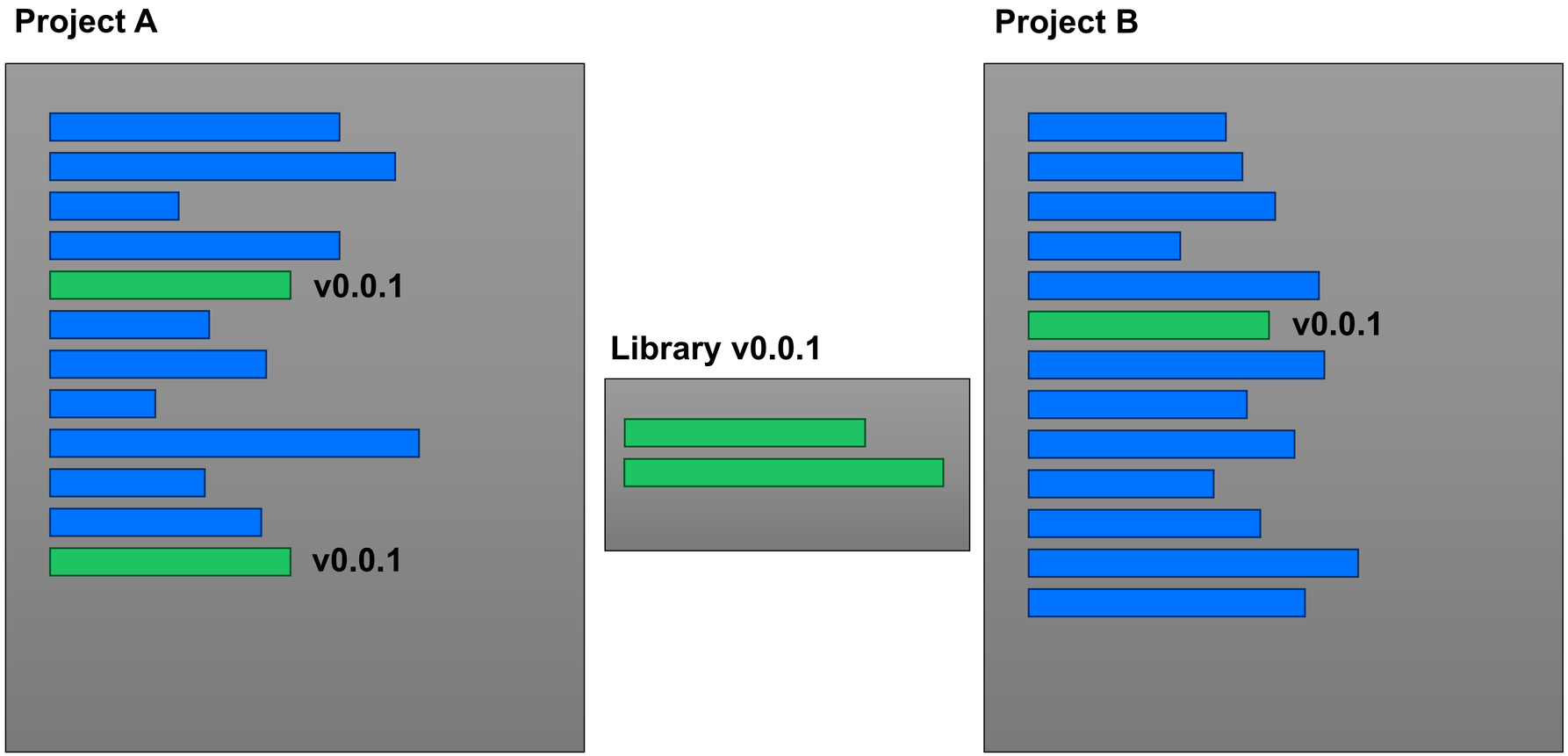
Project A



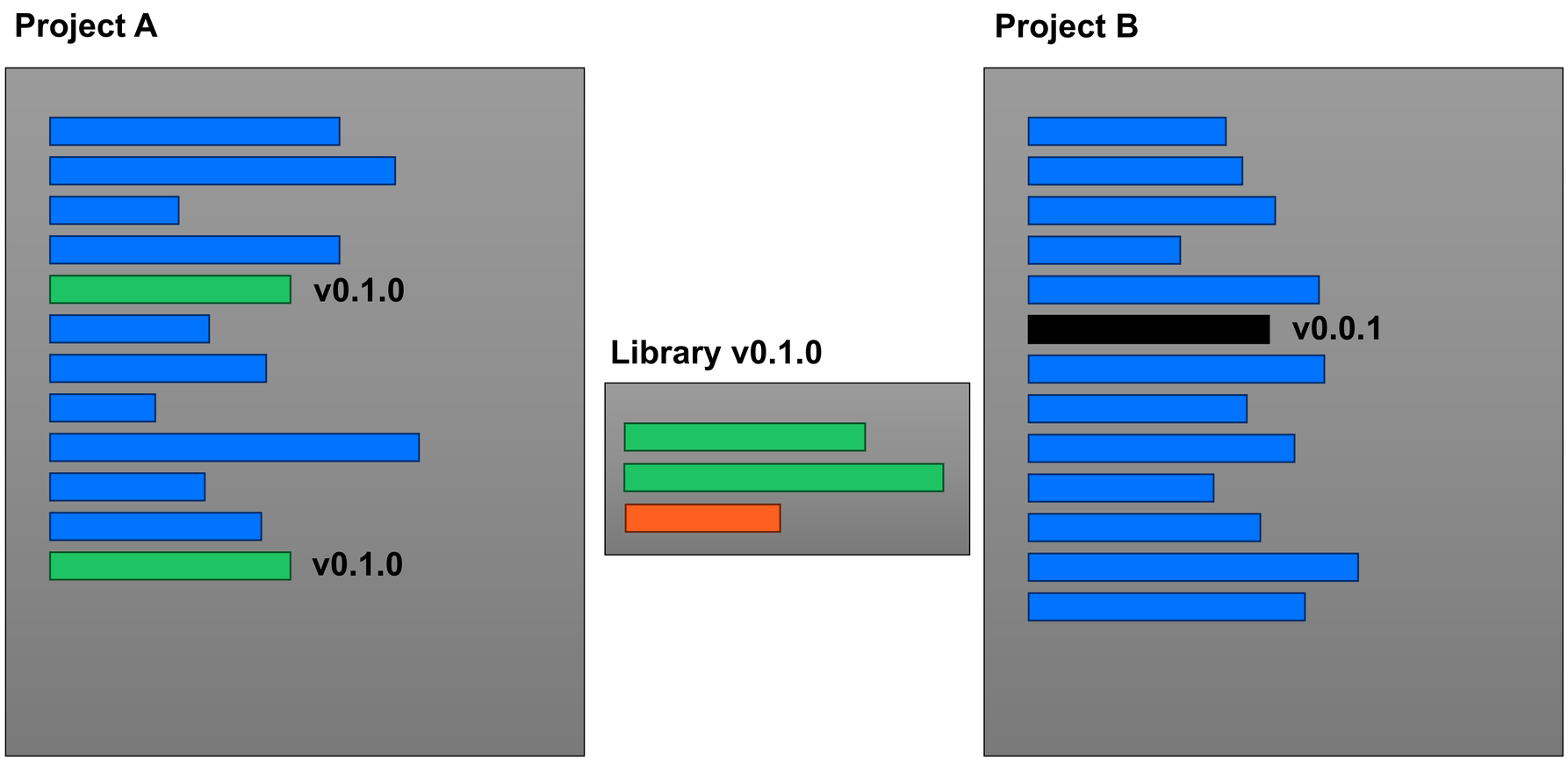
Project B



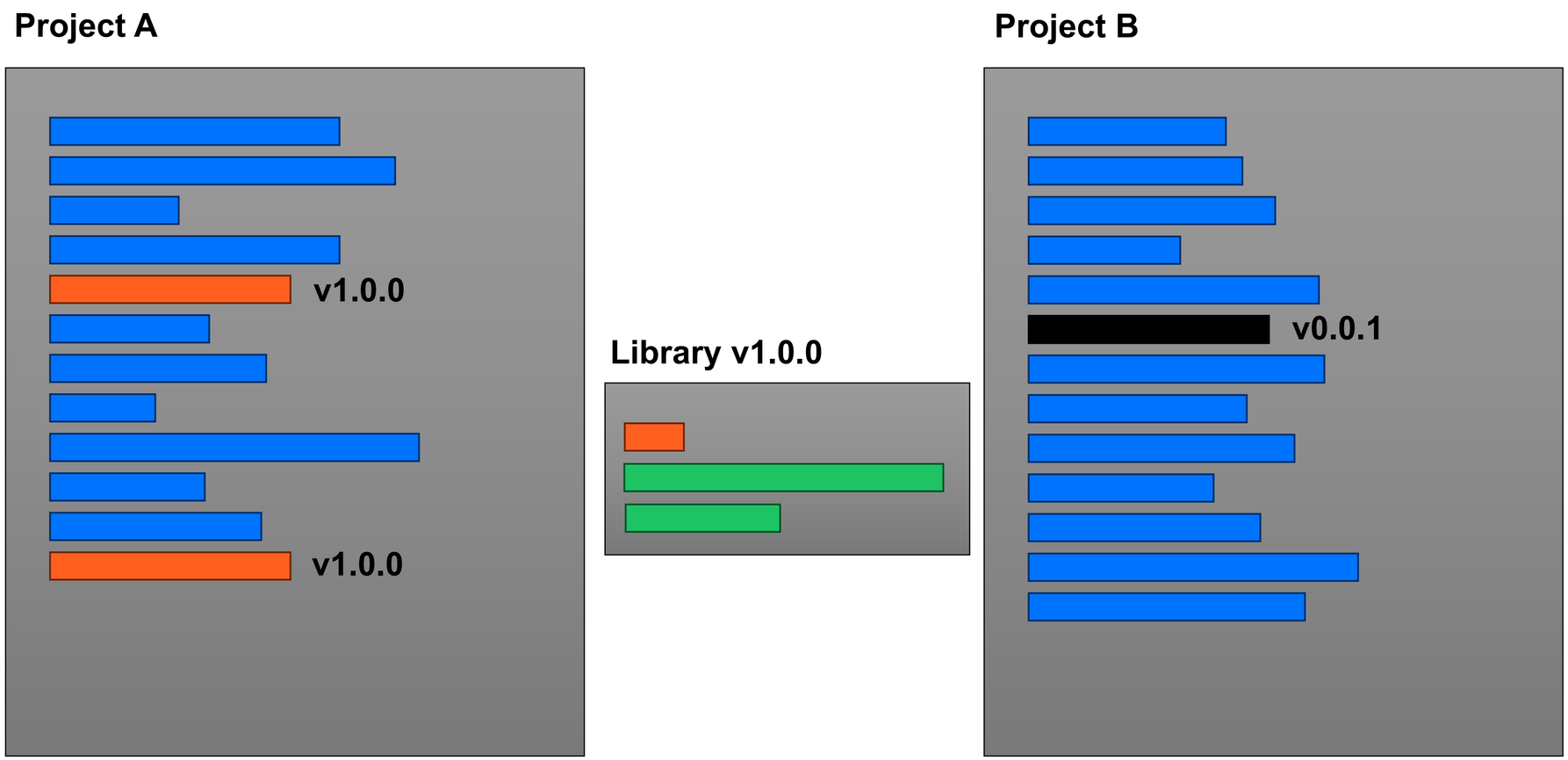
IDENTIFYING A LIBRARY
CREATE A LIBRARY AND MAINTAIN IN ONE PLACE



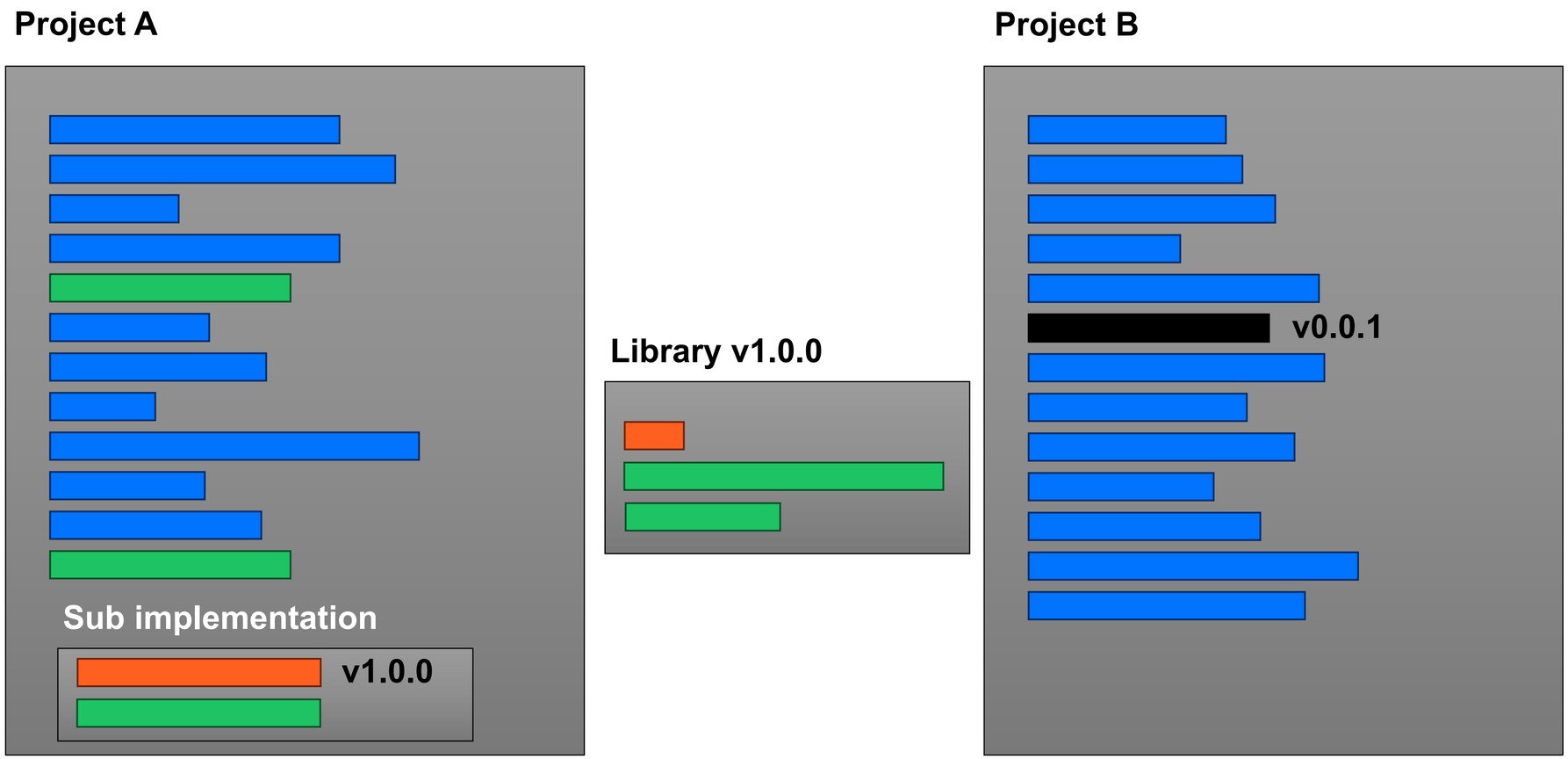
IDENTIFYING A LIBRARY
MINOR UPDATE TO LIBRARY



IDENTIFYING A LIBRARY
MAJOR UPDATE TO LIBRARY



IDENTIFYING A LIBRARY
MAKE YOUR OWN SUB IMPLEMENTATION



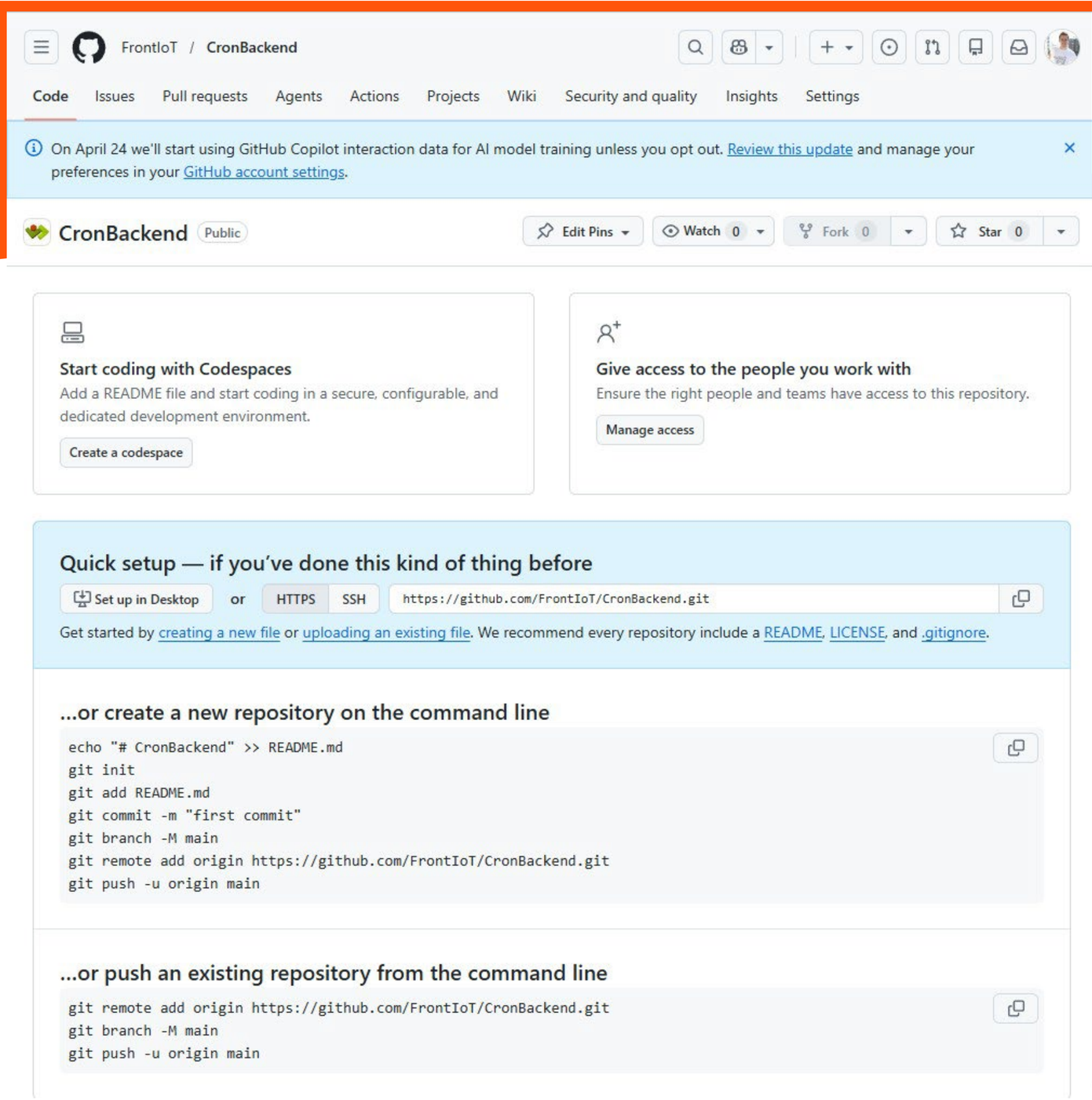
MAKING A LIBRARY

MAKING A LIBRARY
CRONBACKEND EXAMPLE

- The library takes an array of Business Processe handlers and call them at a time set in a cron.txt-file.
- You do not have to compile a new version of the code if you want to stop or run a script at a different time.

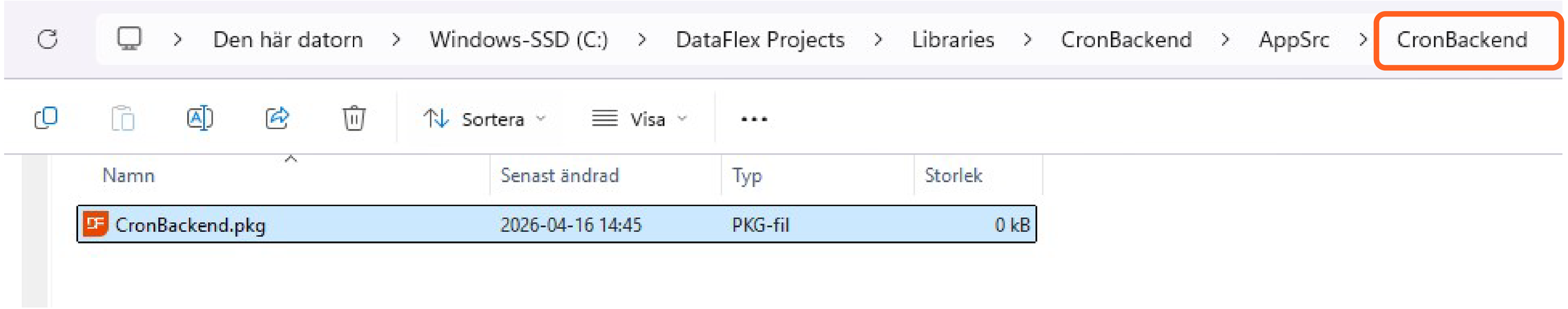
```
# m h dom mon dow oBusinessProcess
 0 * * * * OnceAnHourScript
 0 0 * * * OnceADay
 0 * * * * oSFtpTransfer
 5 * * * * oEDI_Comm
10 * * * * oSFtpTransfer
35 * * * * oEDI_Comm
40 * * * * oSFtpTransfer
```

MAKING A LIBRARY
CREATING A GIT PROJECT



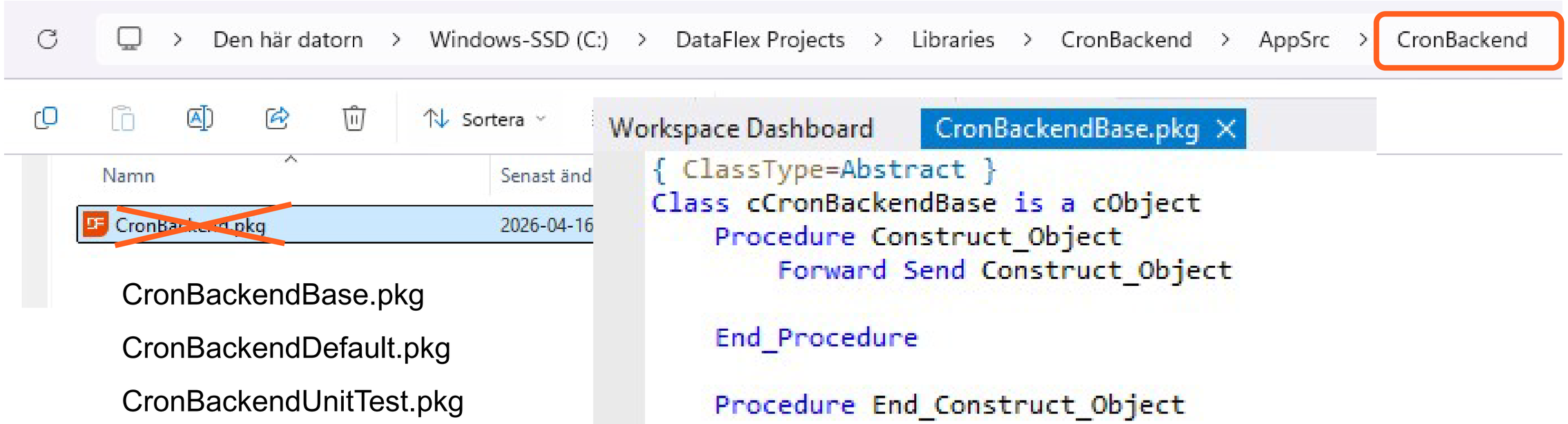
MAKING A LIBRARY
CREATE A WORKSPACE

- Create a blank DF Workspace and create a subfolder to place your library



MAKING A LIBRARY
CREATE A WORKSPACE

- Create a blank DF Workspace and create a subfolder to place your library



MAKING A LIBRARY
AUTOGENERATED SWS-FILE

```
{
    "description": "Managing jobbs that should be executed based on time",
    "df": 26.0
}
```

MAKING A LIBRARY
WORKSPACE SWS FILE WITH MORE INFO

```
{
    "description": "Managing jobbs that should be executed based on time",
    "df": 26.0,
        "package": {
        "df-max": 26.0,
        "df-min": 23.0,
        "icon": "",
        "includeFiles": [
            "CHANGELOG.md"
        ],
        "license": "GPL",
        "licenseFile": "LICENSE",
        "version": "0.0.1"
    },
    "workspaceName": "Cron Backend Scheduler"
}
```

MAKING A LIBRARY
CREATE A BASE CLASS

- Create the base as the access point for your library
- This serves as an abstract clean version

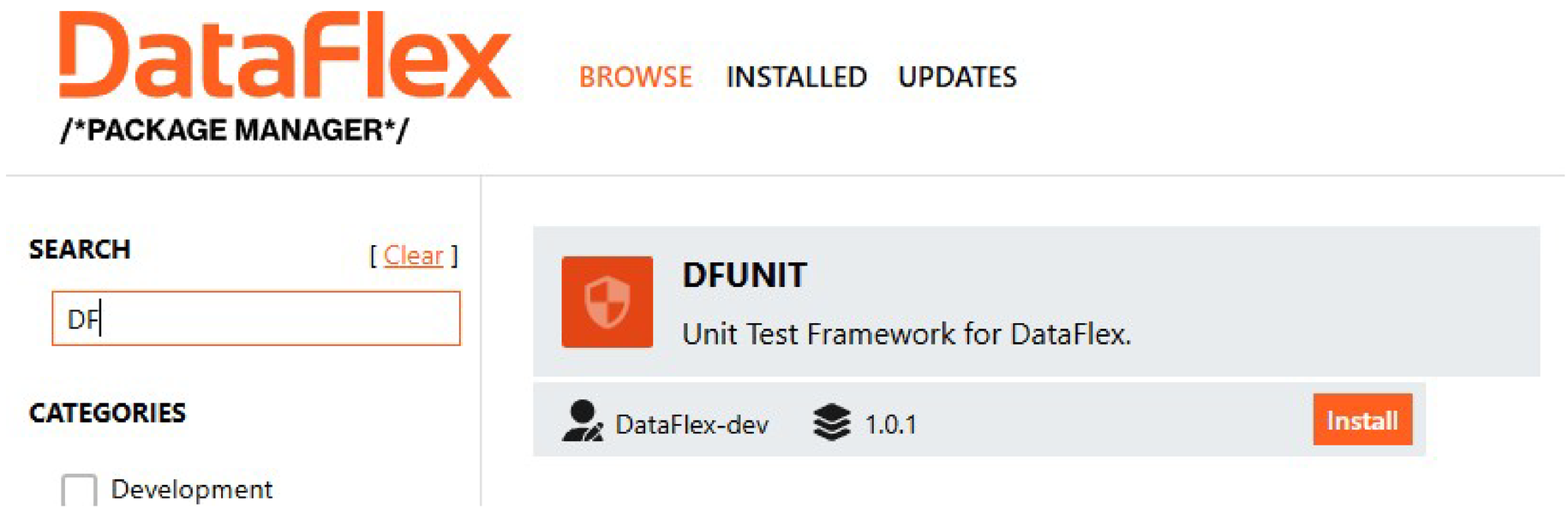
- Create a MyLibraryDefault.pkg that shows a sample
- You can also rely on the UnitTest version as default

Workspace Dashboard | CronBackendBase.pkg

```
{ ClassType=Abstract }
Class cCronBackendBase is a cObject
    Procedure Construct_Object
        Forward Send Construct_Object

    End_Procedure

    Procedure End_Construct_Object

        Forward Send End_Construct_Object
    End_Procedure

    Procedure StartCronProcess

    End_Procedure
End_Class
```

MAKING A LIBRARY
INSTALL DFUNIT

- Find the DF Unit in the Package Manager and add it to your library using the *LibNameUnitTest.sws* workspace



MAKING A LIBRARY
CREATE A UNITTEST PROJECT

Workspace Dashboard | CronBackendBase.pkg | UnitTest.src

```
Use DFUnit.pkg
Use DFAllEnt.pkg
Use cCJStandardCommandBarSystem.pkg

Object oHtmlHelp is a cHtmlHelp
End_Object

Object oApplication is a cApplication
    Set peHelpType to htHtmlHelp

    Object oConnection is a cConnection
        Use LoginEncryption.pkg
        Use DatabaseLoginDialog.dg
    End_Object

End_Object


Object oTestApplication is a cDFUnitTestApplication
    Set pbUseUI to False

End_Object
```

MAKING A LIBRARY
ADD A CHANGELOG FILE

Workspace Dashboard | CronBackendBase.pkg | UnitTest.src | CHANGELOG.md

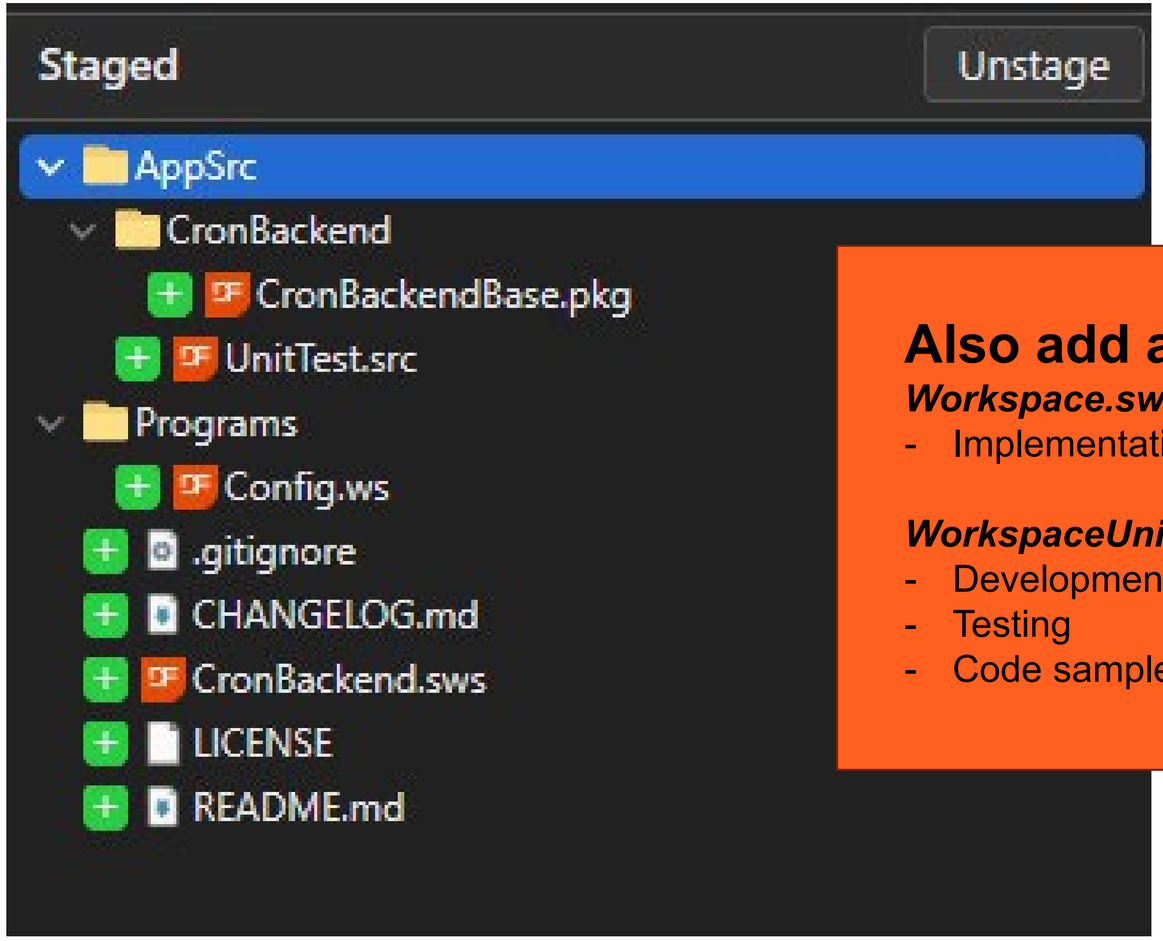
```
# Changelog

All notable changes to this project will be documented in this file.

The format is based on [Keep a Changelog](https://keepachangelog.com/en/1.1.0/),
and this project adheres to [Semantic Versioning](https://semver.org/spec/v2.0.0.html).

## \[Unreleased]

### Changed

## \[0.0.1] - 2026-04-16

* Initial version as a package.
```

MAKING A LIBRARY
ADD A README MD FILE

Workspace Dashboard | CronBackendBase.pkg | UnitTest.src | CHANGELOG.md | README.md

```
# Cron Backend

## Description

    This library creats an easy way to schedule BusinessProcess in your application through a crontab-like configuration file.

## Usage

    Use CronBackend\CronBackendBase.pkg

    Class cCronBackend is a cCronBackendBase
    End_Class

    Object oCronBackend is a cCronBackend
    End_Object

    Send StartCronProcess of oCronBackend
```

MAKING A LIBRARY
PUSH YOUR FIRST VERSION TO GIT



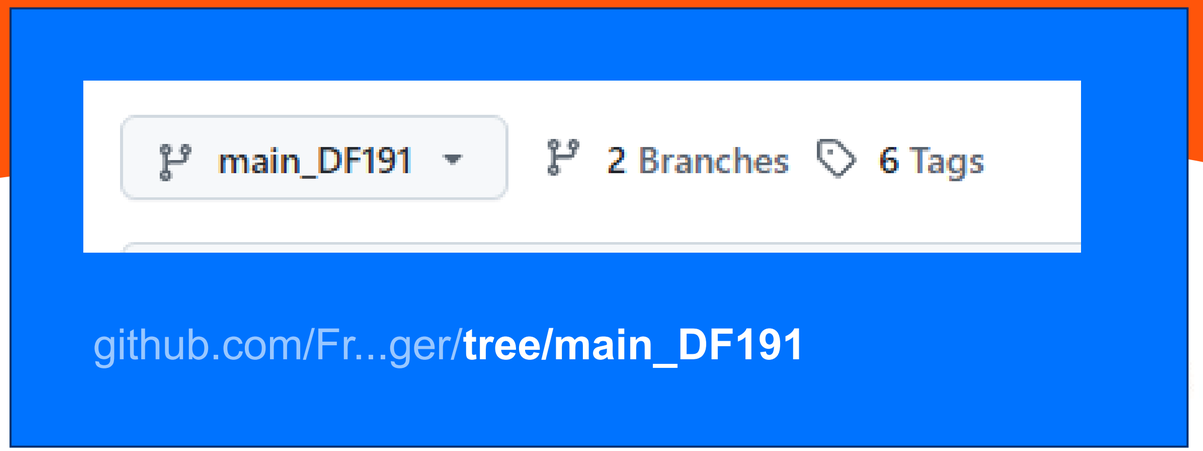
Also add a UnitTest sws-file

Workspace.sws
- Implementations

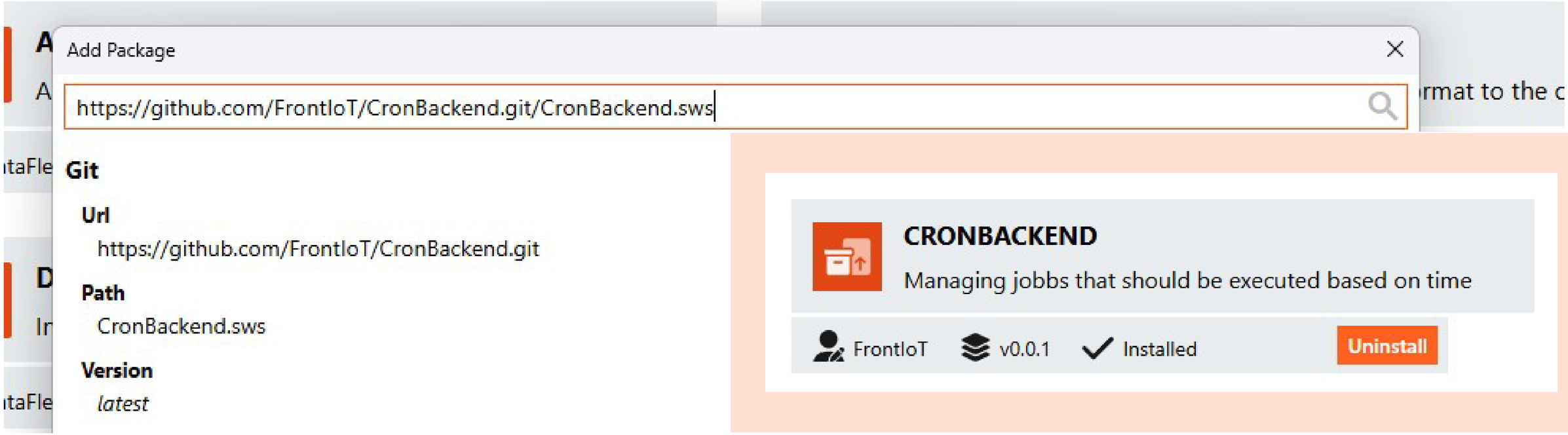
WorkspaceUnitTest.sws
- Development
- Testing
- Code samples

MAKING A LIBRARY
ADD A VERSION TAG

MAKING A LIBRARY
ADD YOUR LIBRARY TO YOUR APPLICATION



NSE INSTALLED UPDATES



MAKING A LIBRARY
CREATE A TEST AND CHECK THAT IT FAILS

Workspace Dashboard | CronBackendBase.pkg | UnitTest.src

```
        Use DatabaseLoginDialog.dg
    End_Object

End_Object

Use CronBackend\CronBackendBase.pkg

Class cCronBackend is a cCronBackendBase
    Procedure Construct_Object
        Forward Send Construct_Object

        Set pbTestState to True
    End_Procedure

End_Class

Object oCronBackend is a cCronBackend
    Property Boolean pbProcessHasStarted False

    Procedure OnProcessStart
        Set pbProcessHasStarted to True
    End_Procedure

End_Object

Object oTestApplication is a cDFUnitTestApplication
    Set pbUseUI to False

    Use UnitTest\Main.pkg

End_Object
```

Main.pkg

```
Object oCronBackendFixtures is a cTestFixture
    Procedure Setup
    End_Procedure

    Procedure TearDown
    End_Procedure

    Object MainStructure is a cTestFixture

        { Published = True }
        Procedure ProcessStarts
            Boolean bProcessHasStarted

            Get pbProcessHasStarted of oCronBackend to bProcessHasStarted
            Send AssertFalse (bProcessHasStarted) 'Process has not started'

            Send StartCronProcess of oCronBackend

            Get pbProcessHasStarted of oCronBackend to bProcessHasStarted
            Send AssertFalse (not(bProcessHasStarted)) 'Process has started'
        End_Procedure
    End_Object

End_Object
```

```
C:\DataFlex Projects\Libraries\CronBackend\Programs>    All Tests
        oCronBackendFixtures
            MainStructure
                ProcessStarts [Failed]
                    Assertion Failed: 'Process has not started'.
                        Should have been [False] but was [True]
------------------------------------------------------------------------
Assertions: 1/2
Tests: 0/1
Errors: 0
------------------------------------------------------------------------
Tests: Failed
```

/*50 YEARS*/

MAKING A LIBRARY
UPDATE THE DOCUMENTATION

Workspace Dashboard | CronBackendBase.pkg | UnitTest.src | CHANGELOG.md | README.md

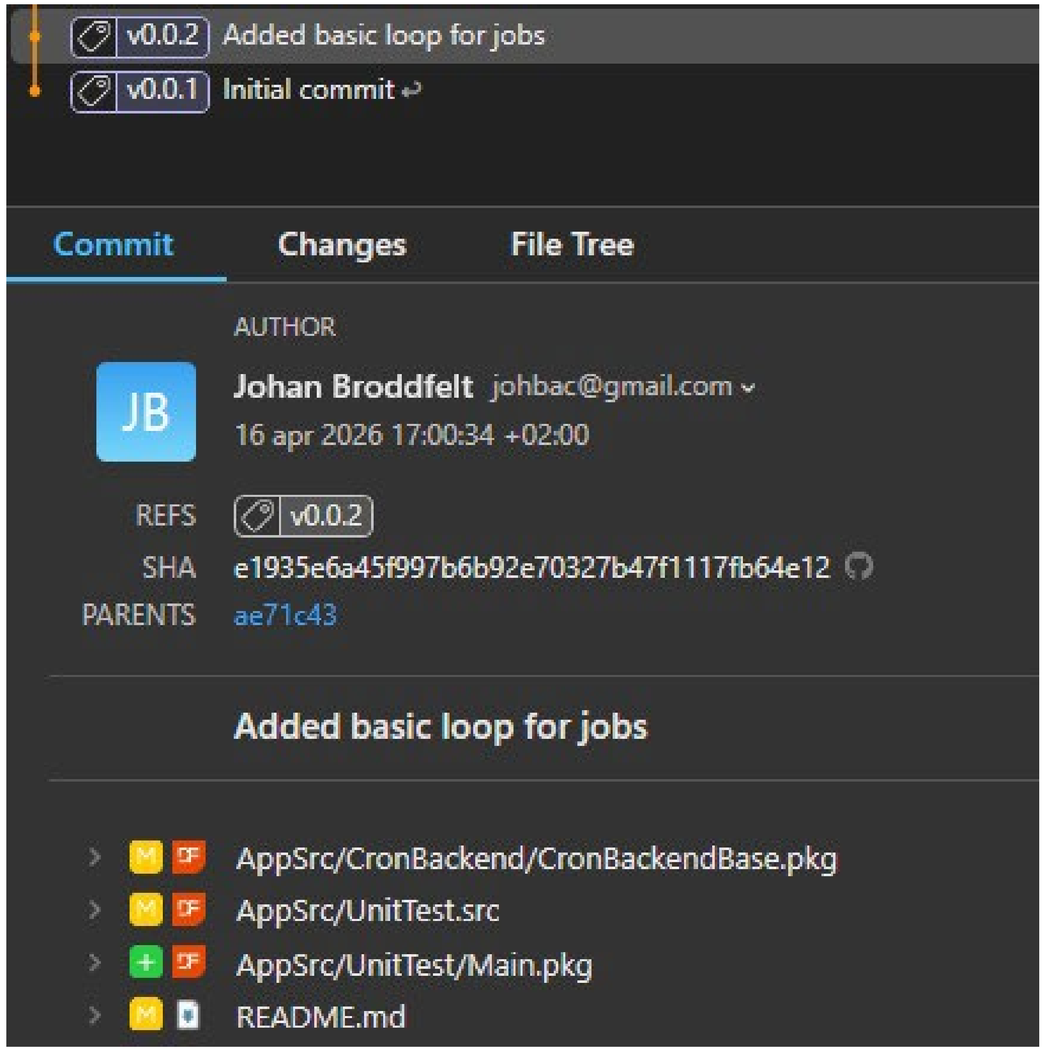
```
# Cron Backend

## Description

    This library creats an easy way to schedule BusinessProcess in your application through a crontab-like configuration

## Usage

    Use CronBackend\CronBackendBase.pkg

    Class cCronBackend is a cCronBackendBase
        Procedure OnProcessStart
            // Here is a good place to put tests calls of your Business Processes to run directly during development
        End_Procedure
    End_Class

    Object oCronBackend is a cCronBackend
    End_Object

    Send StartCronProcess of oCronBackend


## Properties

        Boolean pbTestState False // Primarilly used for UnitTest. Only run through the loop once
        Integer piIntervalFrequency 5 // Seconds between cycles
        Integer piRestartAfterCycles (12 * 60) // 12 * 5 seconds * 60 => 1 hour
        Boolean pbExitAfterRestartAfterCycles False // Use this if you have issues with memory leaks and configure the Ta
```

MAKING A LIBRARY
UPDATE CHANGELOG

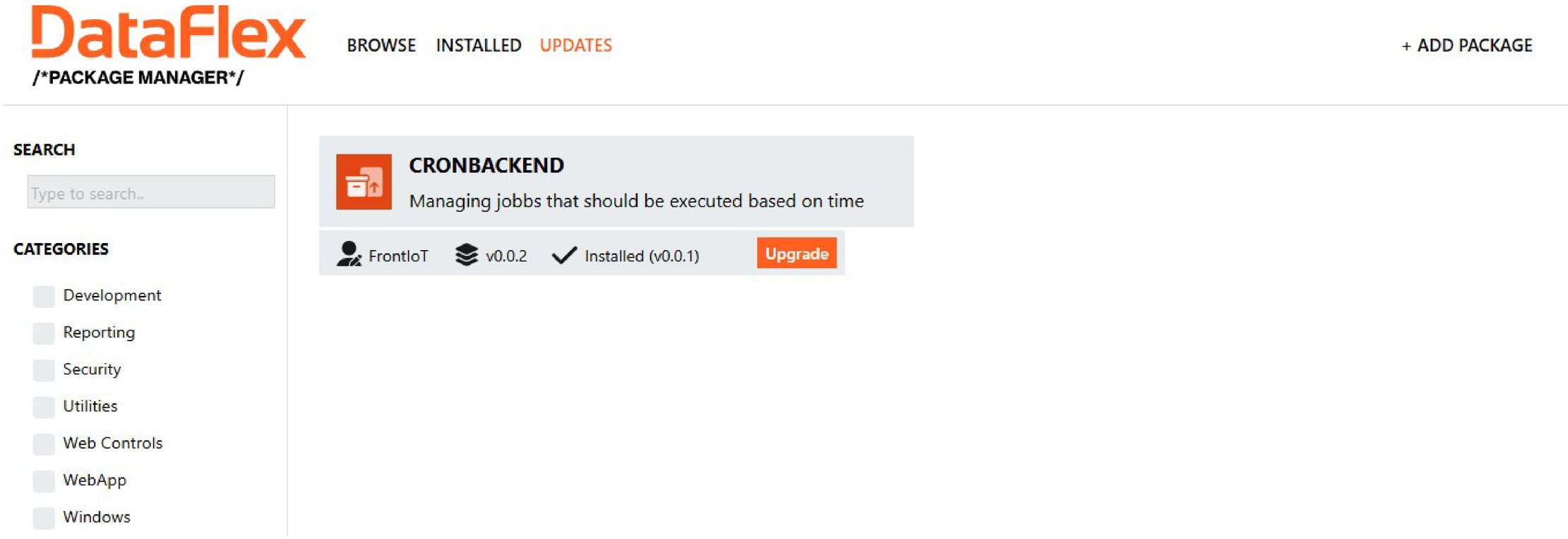
Workspace Dashboard | CronBackendBase.pkg | UnitTest.src | CHANGELOG.md | README.md

```
# Changelog

All notable changes to this project will be documented in this file.

The format is based on [Keep a Changelog](https://keepachangelog.com/en/1.1.0/),
and this project adheres to [Semantic Versioning](https://semver.org/spec/v2.0.0.html).

## \[Unreleased]

### Changed

## \[0.0.2] - 2026-04-16

* Added the basic loop with test option

## \[0.0.1] - 2026-04-16

* Initial version as a package.
```

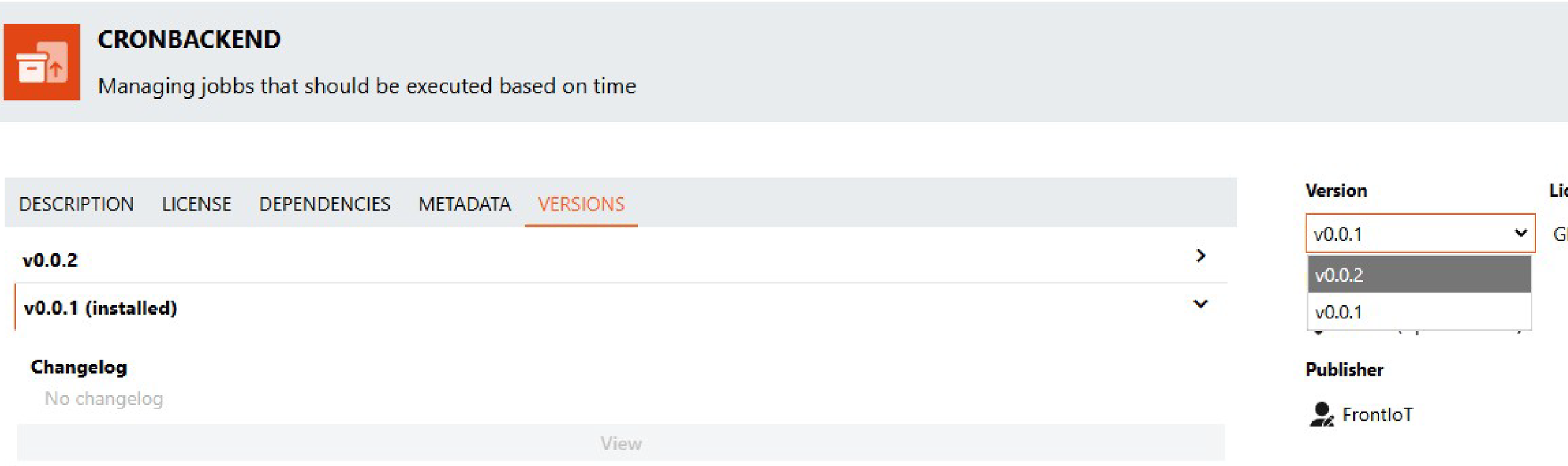
MAKING A LIBRARY
PUSH THE CHANGES TO GIT



MAKING A LIBRARY
UPGRADE USING THE BUTTON…

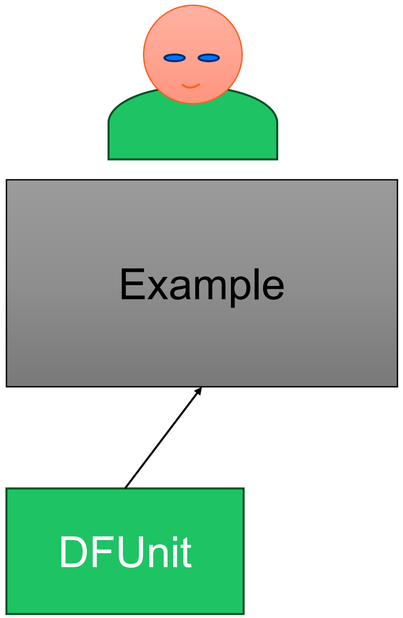


MAKING A LIBRARY
… OR UPDATE INSIDE THE PACKAGE USING THE VERSION DROPDOWN



MAKING A LIBRARY
SAMPLE LIBRARIES

https://github.com/frontiot/



CronBackend

Starts BP:s at a crontab like schedule

https://github.com/FrontIoT/CronBackend

Does

- Loop array of BP:s created by you
- Read *cron.txt* file to know when

Does not

- Have any predefined BP:s other then for Unit Test

VersionUpdateManager

Run version update functions to manage none-code changes

https://github.com/FrontIoT/VersionUpdateManager

Does

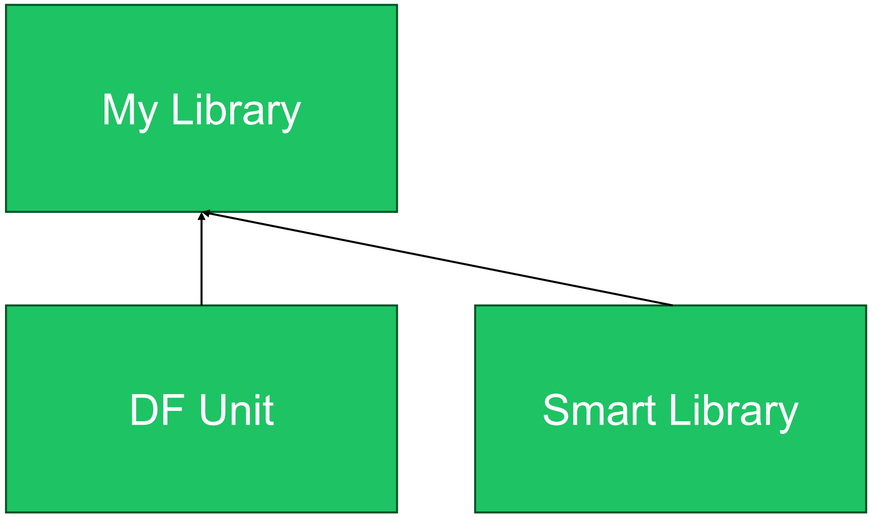
- Ask your system for current version *(integer)*
- Loop array with class references to updates you create

Does not

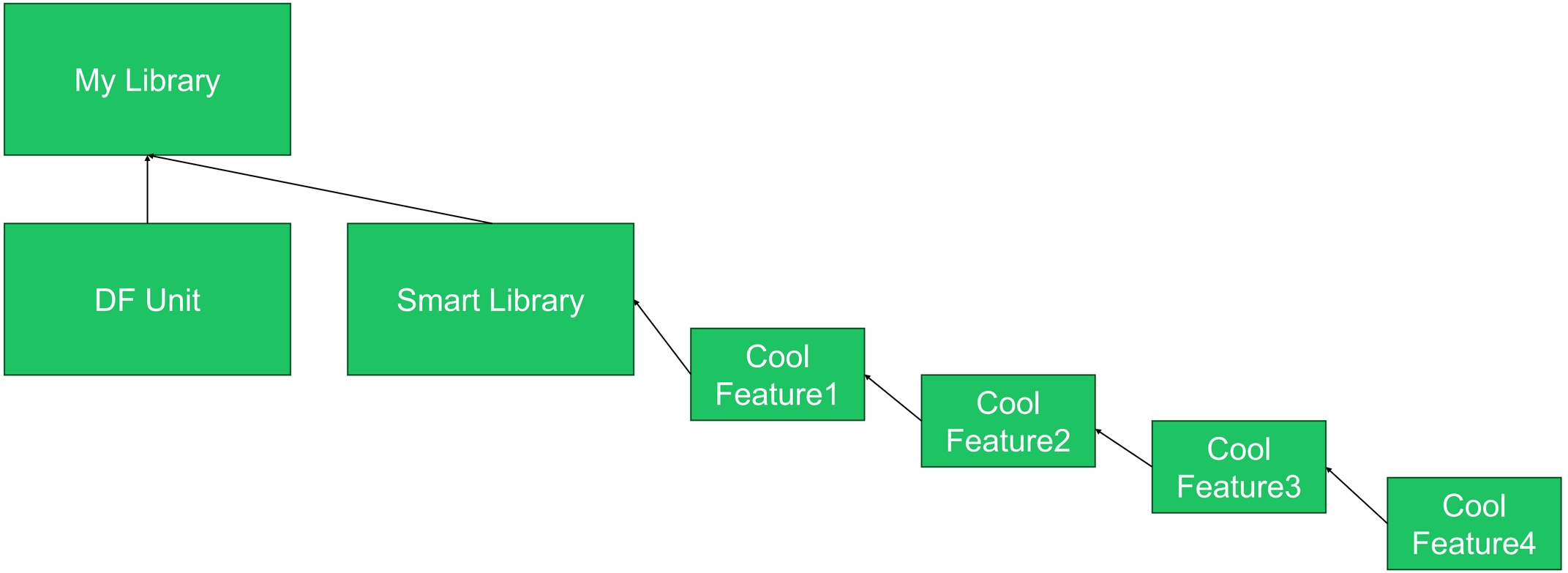
- Store version numbers
- Contain code to update the database*

MANAGE DEPENDENCIES

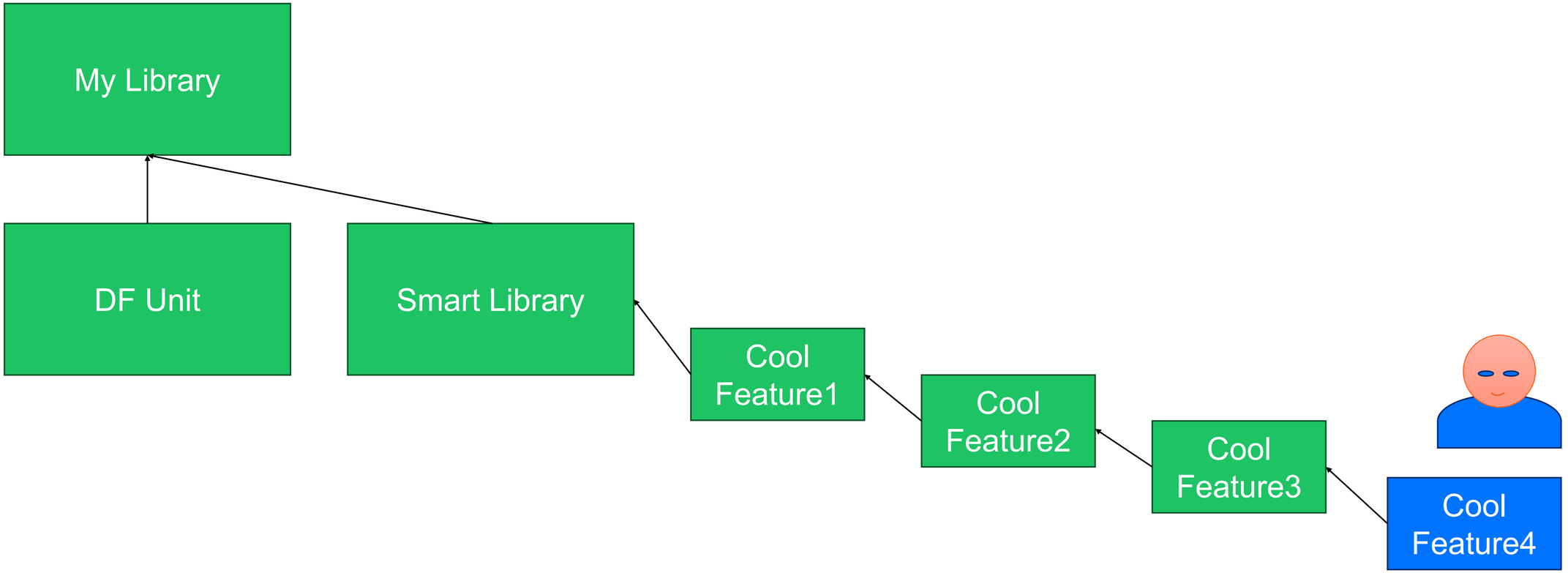
MANAGE DEPENDENCIES
DEPENDENCIES



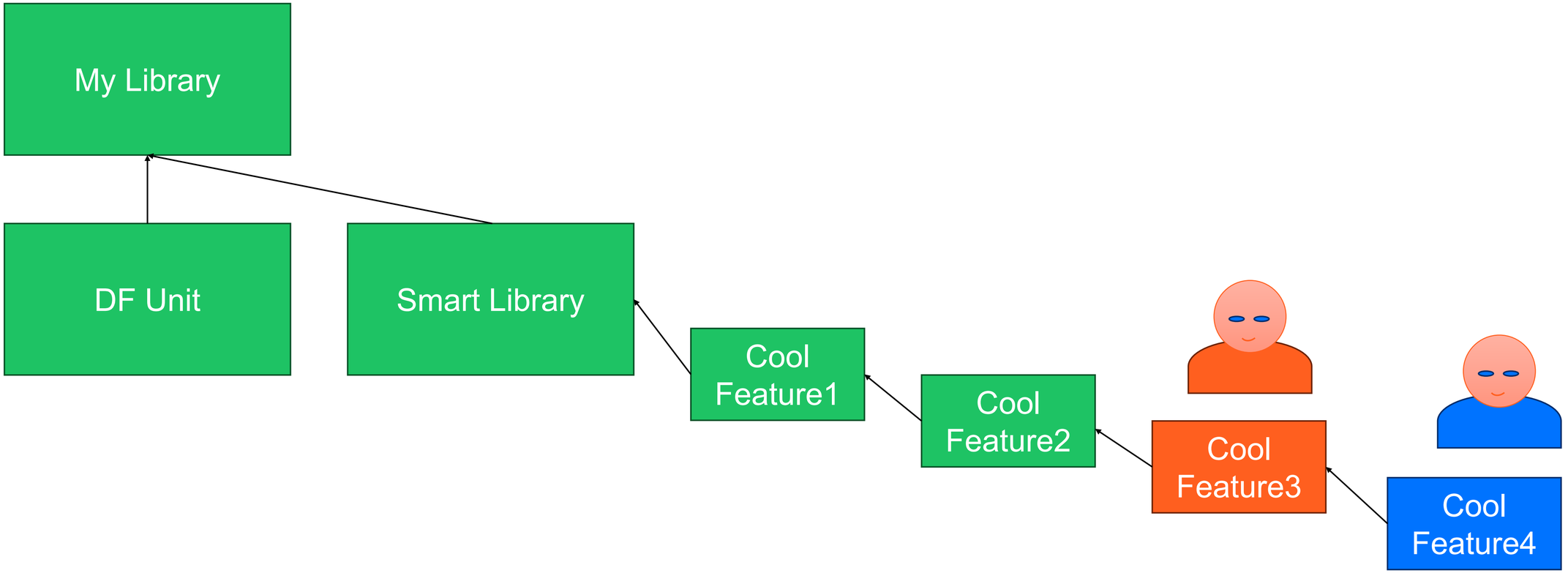
MANAGE DEPENDENCIES
MULTIPLE DEPENDENCIES



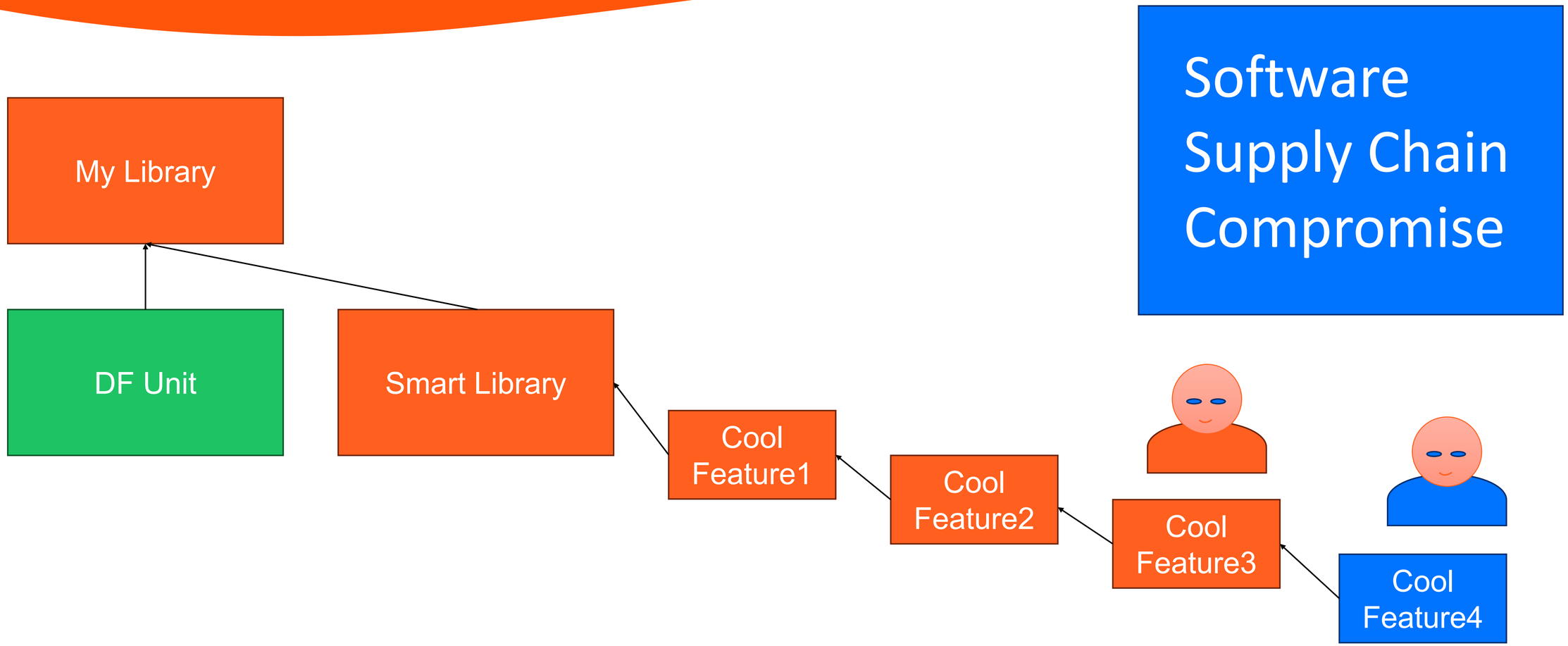
MANAGE DEPENDENCIES
HARDER TO STAY UP TO DATE



MANAGE DEPENDENCIES
TRUST IN MORE PARTIES

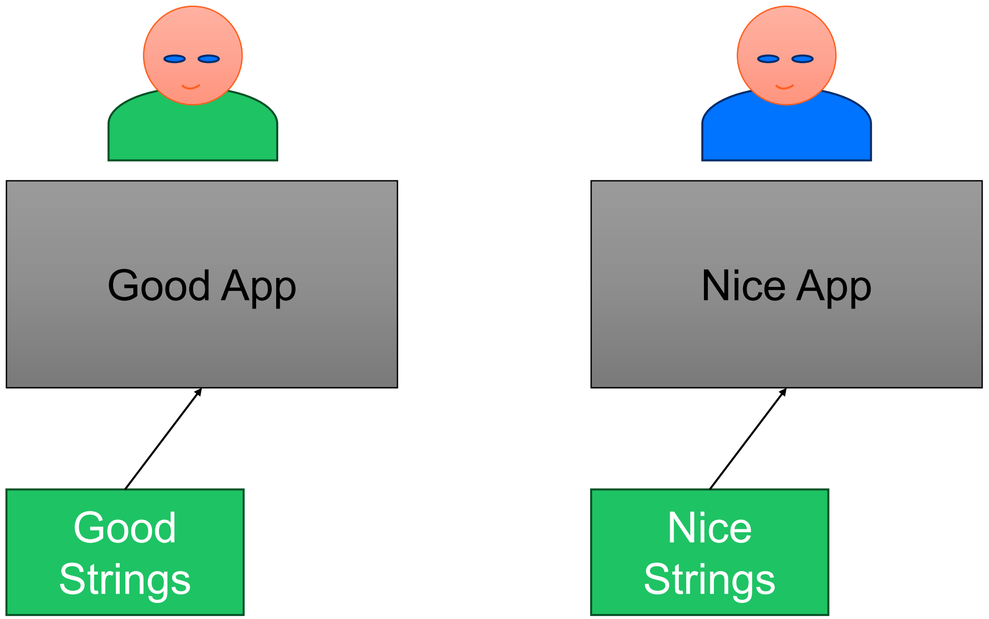


MANAGE DEPENDENCIES
RISK OF INVALID OR INSECURE CODE

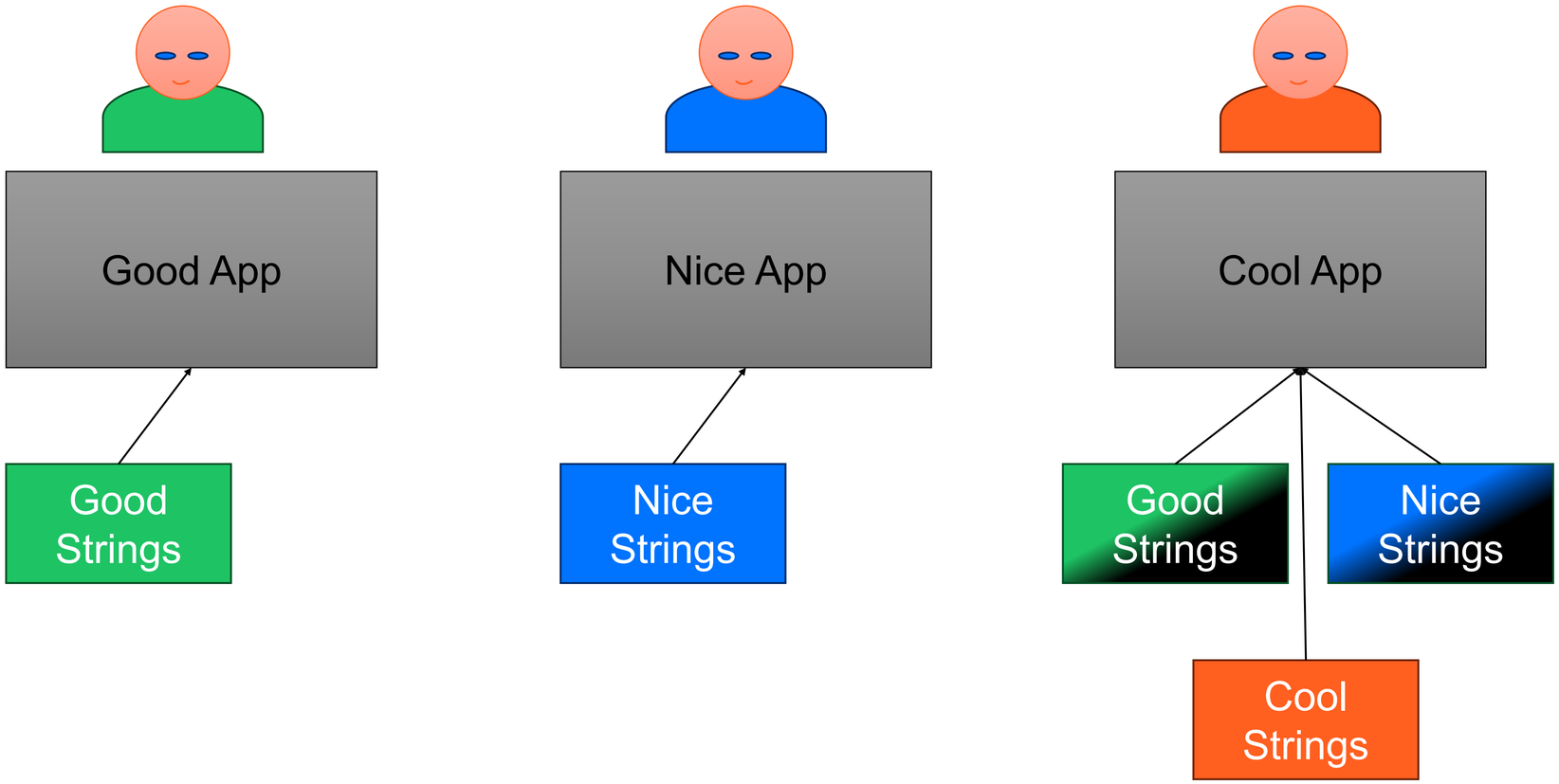


COMMUNITY LIBRARIES

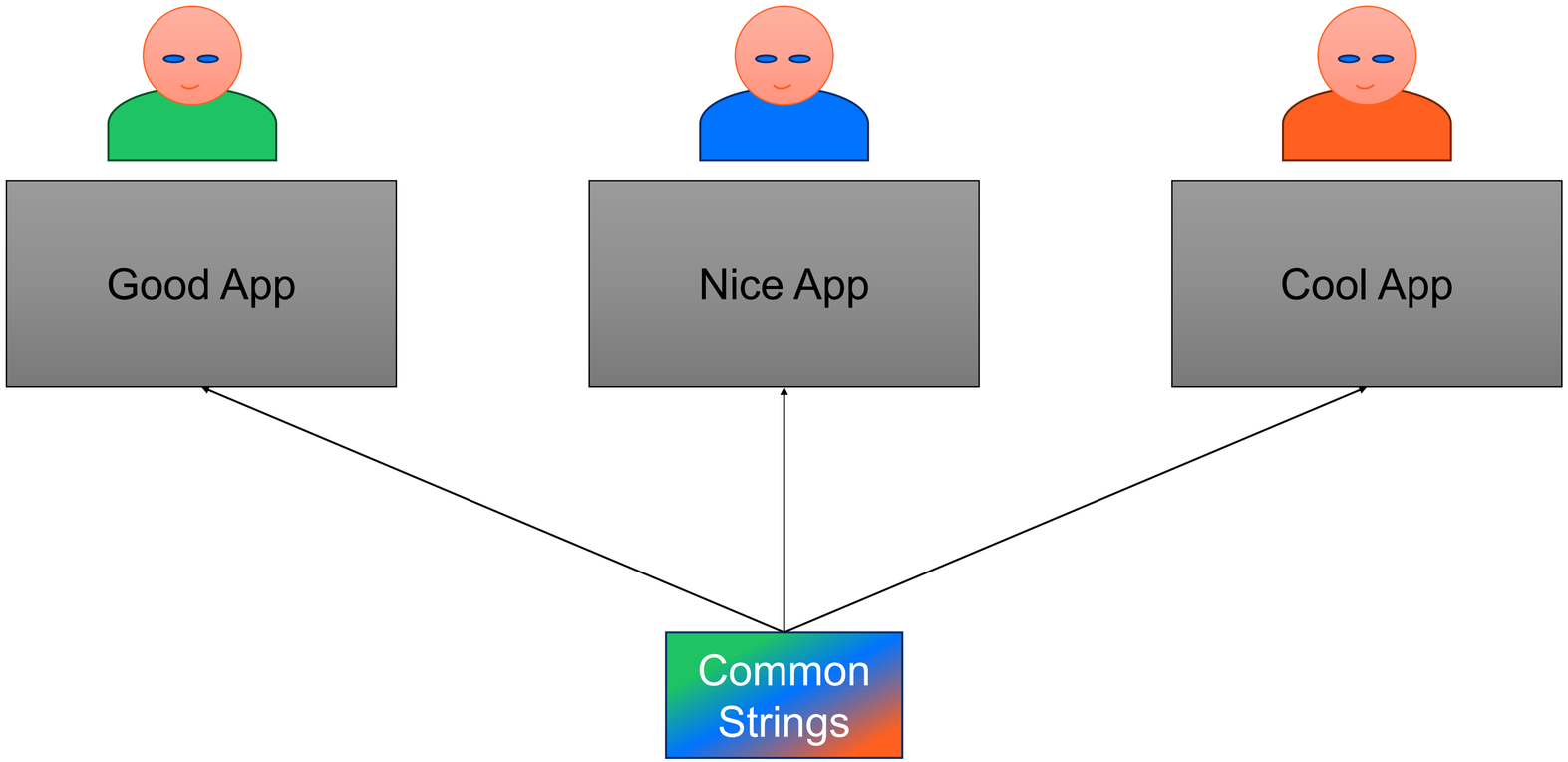
COMMUNITY LIBRARIES
MAINTAINING YOUR OWN VERSION OF COMMON FUNCTIONS



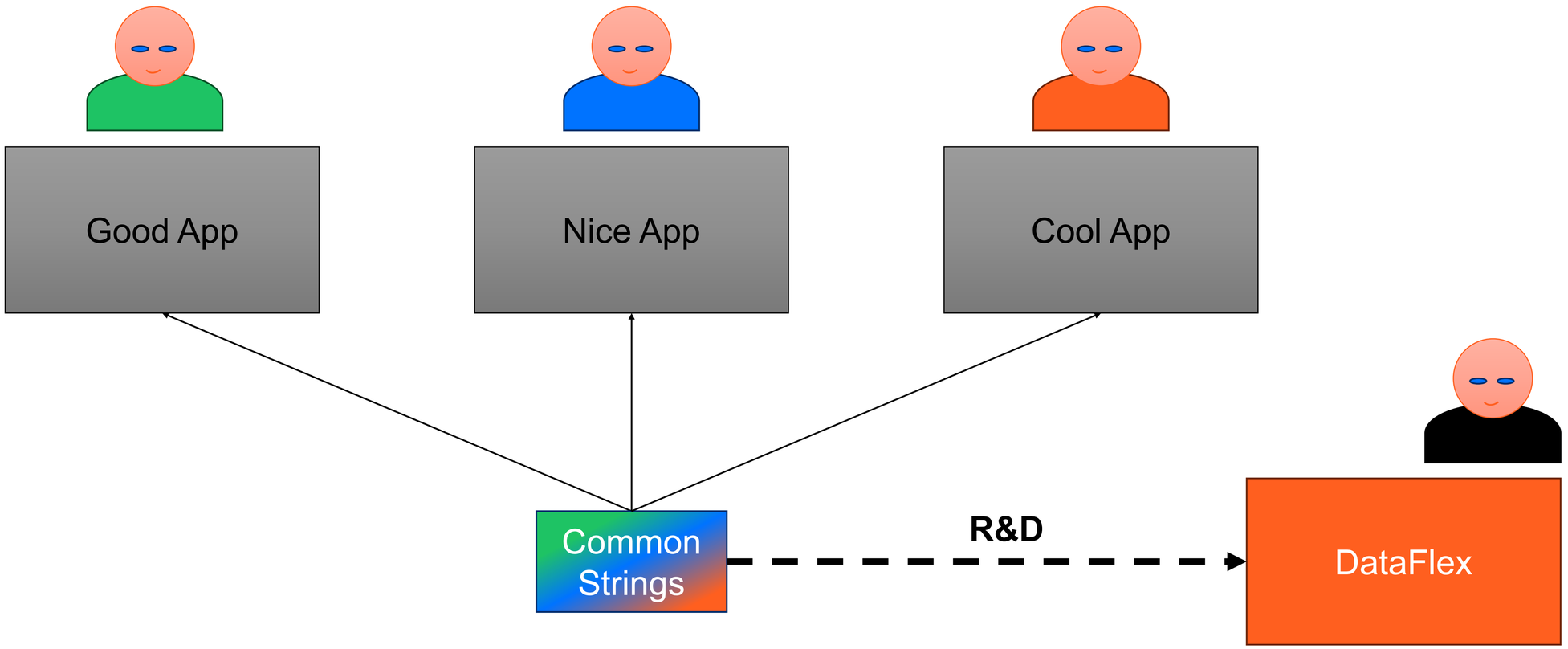
COMMUNITY LIBRARIES
NOW WE HAVE A MESS OF HALF-BAKED LIBRARIES



COMMUNITY LIBRARIES
COMMON COMMUNITY LIBRARY CAN SOLVE THE BASICS



COMMUNITY LIBRARIES
USEFUL FEATURES CAN BE INTEGRATED IN DATAFLEX



VERSION MANAGEMENT

VERSION MANAGEMENT
SEMANTIC VERSIONING (SEMVER)

Major.Minor.Patch v0.0.0

- Major: Change the signature or remove obsolete code
- Minor: New feature or added parameter with defaults
- Patch: Fix a bug or optimize

Pre-release (-) and build (+) notation

- 1.0.0-alpha
- 1.0.0-beta
- 1.0.0-rc.1
- *1.0.0+note_about_this_releas*

Patch v0.0.1

Fix a bug, improve speed, optimize in other ways

This should NOT affect how the library is used or the values it returns

I thought this update would be a Patch.

from: **Move sVal to asVals[SizeOfArray(asVals)]**

to: **Move sVal to asVals[-1]**

But it broke an implementation in **DF23**. So, it is more like a Major.

SWS: "df-min": 25.0

Minor v0.1.0

New feature, additional parameter with a default value.

This should not change any of the existing behavior but does add functionality.

keep: Procedure DoStuff String sData

add: Procedure DoStuff**V2** String sData String sExtra

or: Procedure DoStuff String sData String sExtra

    If (Num_Arguments < 2) Move " " to sExtra

Major v1.0.0

Changes the signature of a function or remove deprecated features.

If you can do a Minor, it is often preferred.

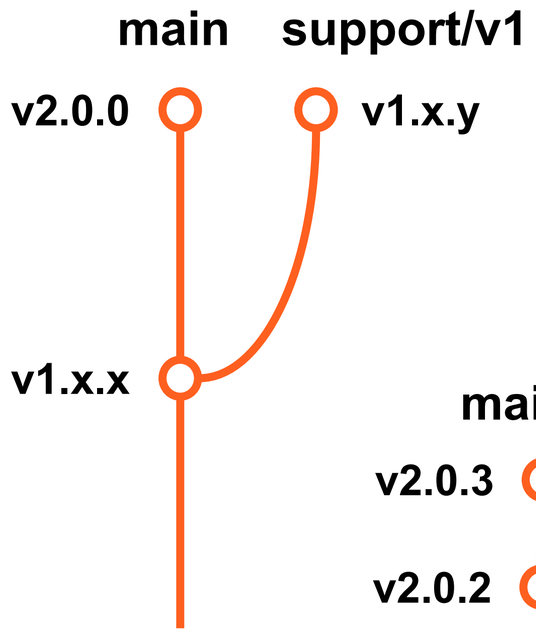
NOTE: If a feature is to be removed it must have been marked as **{ Obsolete=True }** for at least one Major release cycle.

v2.x.y *marked as Obsolete*

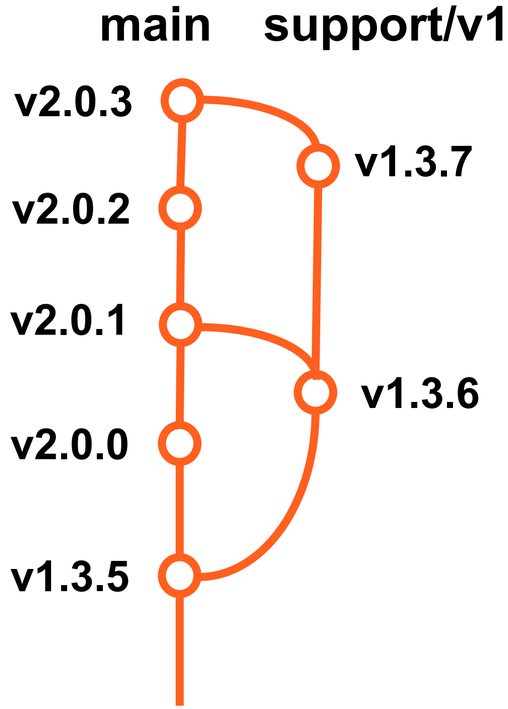
v3.x.y *stay as Obsolete*

v4.0.0 *Removed*

VERSION MANAGEMENT
MAJOR VERSIONS AND THE LONG-TAIL SUPPORT (LTS)



- Before releasing v2.0.0
  Create a new branch from your main code called **support/v1**
- Main branch get tagged as v2.0.0
  Remove obsolete functions and change behavior
- Fix bugs and patches in support/v1 branch
  Push v1.0.1 and so on here
- Switch back to main v2.0.0 and update from v1.0.0
  Use Git's **cherry-pick** or **merge** to keep v2 up to date with the same patches



/*50 YEARS*/

GOOD LIBRARY PRACTICES

GOOD LIBRARY PRACTICES
LICENSE

MIT

- Use it however you want.

Apache 2.0

- Use it however you want, but you may not patent my code as your own.

LGPL

- Use it in proprietary software, but improvements to the library itself should remain open.

Unlicense

- Considered public domain

Other common licenses: *BSD 2-Clause, BSD 3-Clause, GPL 2.0, GPL 3.0, AGPL 3.0, MPL 2.0*

Documentation

Clear description of **what the library does** and how to use it

Unit Tests

Insurance Policy that in one execution verifies that the library is still working

It also serves as a great sample of how to use the library

Versions

Informs the developer **what to expect when upgrading** the package

No global variables

Everything should be contained and testable

Name prefixing

Prefix classes and methods with the library name or an abbreviation of it

Use **MyLib_ConcatString** instead of just **ConcatString** to avoid conflicts

Separate Unit Test workspace

Keep the implementation workspace clean from Unit Test to avoid conflicts with duplicat versions of the DFUnit library if it is used in the application as well

Need only one method from another library

Sometimes it is ok to copy a function for internal use in your library to avoid dependency.

Build abstract class but provide default

The abstract class contains the base logic, while the default shows a basic implementation. { ClassType=Abstract }

Private

Use { Visibility=Private } to indicate that the method should only be used by the library

/* QUESTIONS? */

EXAMPLES AT
GIT: https://github.com/FrontIoT

REACH ME AT
MAIL: jb@frontiot.com