

Technological foundations of software development

Document, license, publish, maintain your software

ICM – Computer Science Major – Course unit on Technological foundations of computer science

M1 Cyber Physical and Social Systems – Course unit on CPS2 engineering and development, Part 2: Technological foundations of software development

Maxime Lefrançois <https://maxime-lefrancois.info>

online: <https://ci.mines-stetienne.fr/cps2/course/tfsd/>

Objectives of the session

This session aims to familiarize you with the methods and tools for Document, license, publish, maintain your software.
In particular, we will cover tools for Java, Python, JavaScript.

Technological foundations of software development

Document, license, publish, maintain your software

Part 1: Documenting your code

ICM – Computer Science Major – Course unit on Technological foundations of computer science

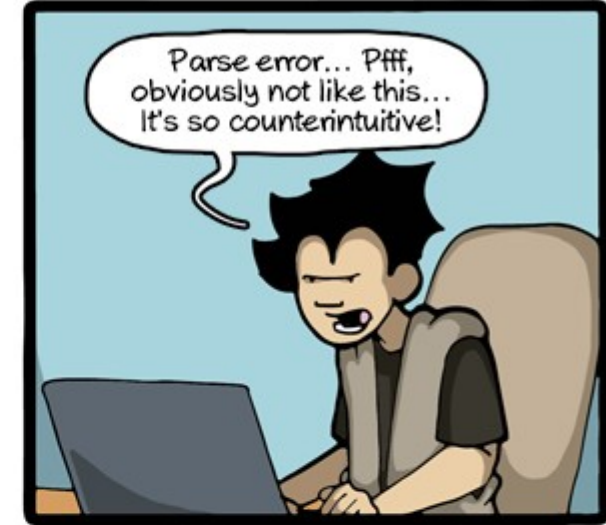
M1 Cyber Physical and Social Systems – Course unit on CPS2 engineering and development, Part 2: Technological foundations of software development

Maxime Lefrançois <https://maxime-lefrancois.info>

online: <https://ci.mines-stetienne.fr/cps2/course/tfsd/>

Documenting your code: Why ?

- Explain how the program works
- Explain how to use the program



Types of documentation

- **Requirements** - Statements that identify the attributes, capabilities, characteristics or qualities of a system. It is the basis for what will be or has been implemented.
- **Architecture/Design** - An overview of the software. Includes the relationships with an environment and the construction principles to be used in the design of software components.
- **Technical** - Documentation of code, algorithms, interfaces and APIs.
- **End User** - Manuals for the end user, system administrators, and support staff.
- **Marketing** - How to market the product and market demand analysis.

Requirements specification document

specification (in software engineering)

“production of a document that can be systematically reviewed, evaluated, and approved”

— ISO/IEC TR 19759:2015 Software Engineering Body of Knowledge (SWEBOK)

software requirements specification (SRS)

“structured collection of the essential requirements [functions, performance, design constraints and attributes] of the software and its external interfaces”

— IEEE 1012-2016 - IEEE Standard for System, Software, and Hardware Verification and Validation

Structure [edit]

An example organization of an SRS is as follows:^[6]

- 1. Purpose
 - 1. Definitions
 - 2. Background
 - 3. System overview
 - 4. References
- 2. Overall description
 - 1. Product perspective
 - 1. System Interfaces
 - 2. User interfaces
 - 3. Hardware interfaces
 - 4. Software interfaces
 - 5. Communication Interfaces
 - 6. Memory constraints
 - 2. Design constraints
 - 1. Operations
 - 2. Site adaptation requirements
 - 3. Product functions
 - 4. User characteristics
 - 5. Constraints, assumptions and dependencies
- 3. Specific requirements
 - 1. External interface requirements
 - 2. Performance requirements
 - 3. Logical database requirement
 - 4. Software system attributes
 - 1. Reliability
 - 2. Availability
 - 3. Security
 - 4. Maintainability
 - 5. Portability
 - 5. Functional requirements
 - 1. Functional partitioning
 - 2. Functional description
 - 3. Control description
 - 6. Environment characteristics
 - 1. Hardware
 - 2. Peripherals
 - 3. Users
- 7. Other

example organization of a SRS document

source: https://en.wikipedia.org/wiki/Software_requirements_specification

Architecture/design

Software architecture description

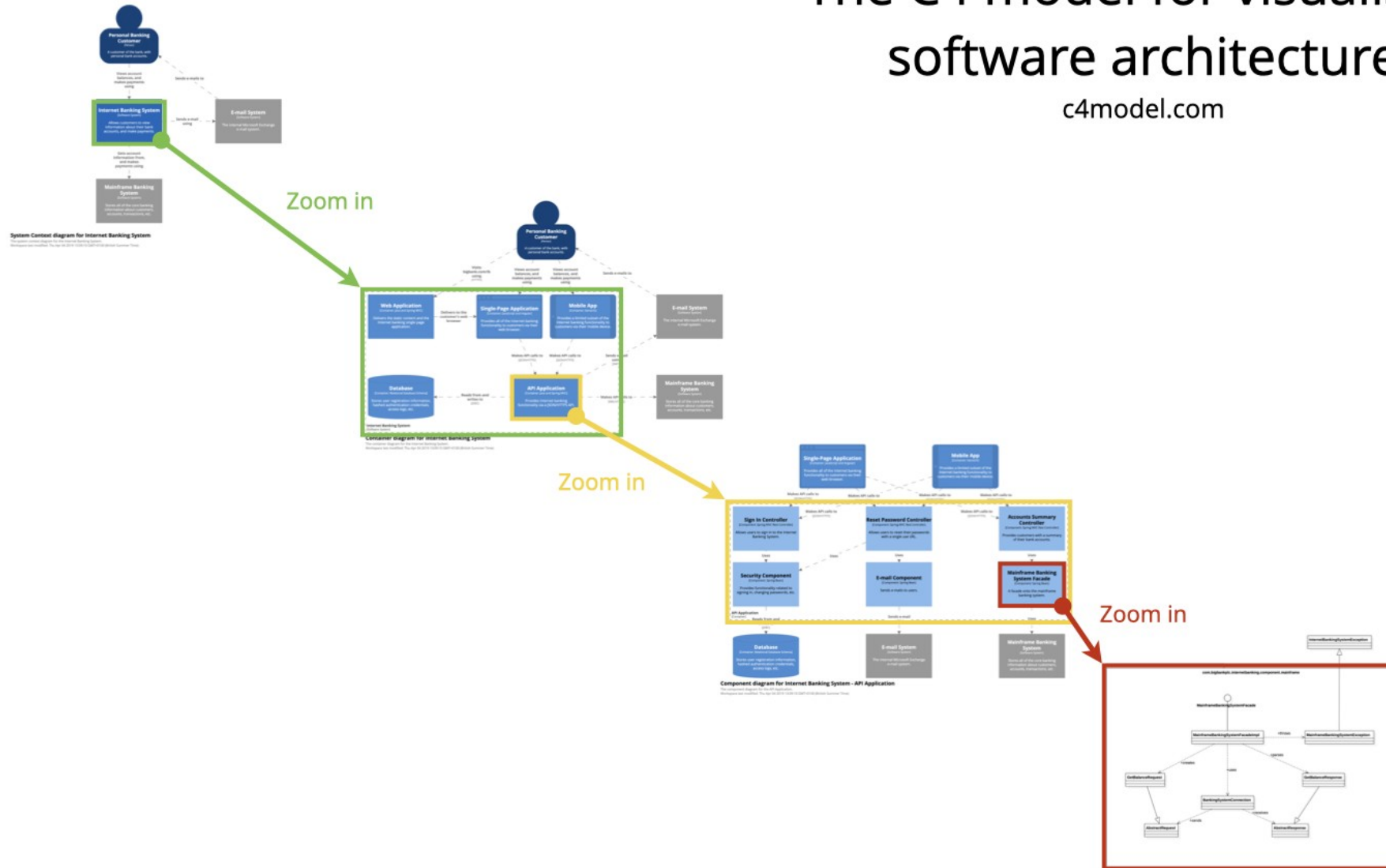
the set of practices for expressing, communicating and analysing [software architectures](#) (also called architectural rendering), and the result of applying such practices through a work product expressing a software architecture

— ISO/IEC/IEEE 42010 Systems and software engineering — Architecture description

existing languages such as the UML can be used as Architecture description languages for analysis, design, and implementation of software-based systems as well as for modeling business and similar processes.

The C4 model for visualising software architecture

c4model.com



Level 1
Context

Level 2
Containers

Level 3
Components

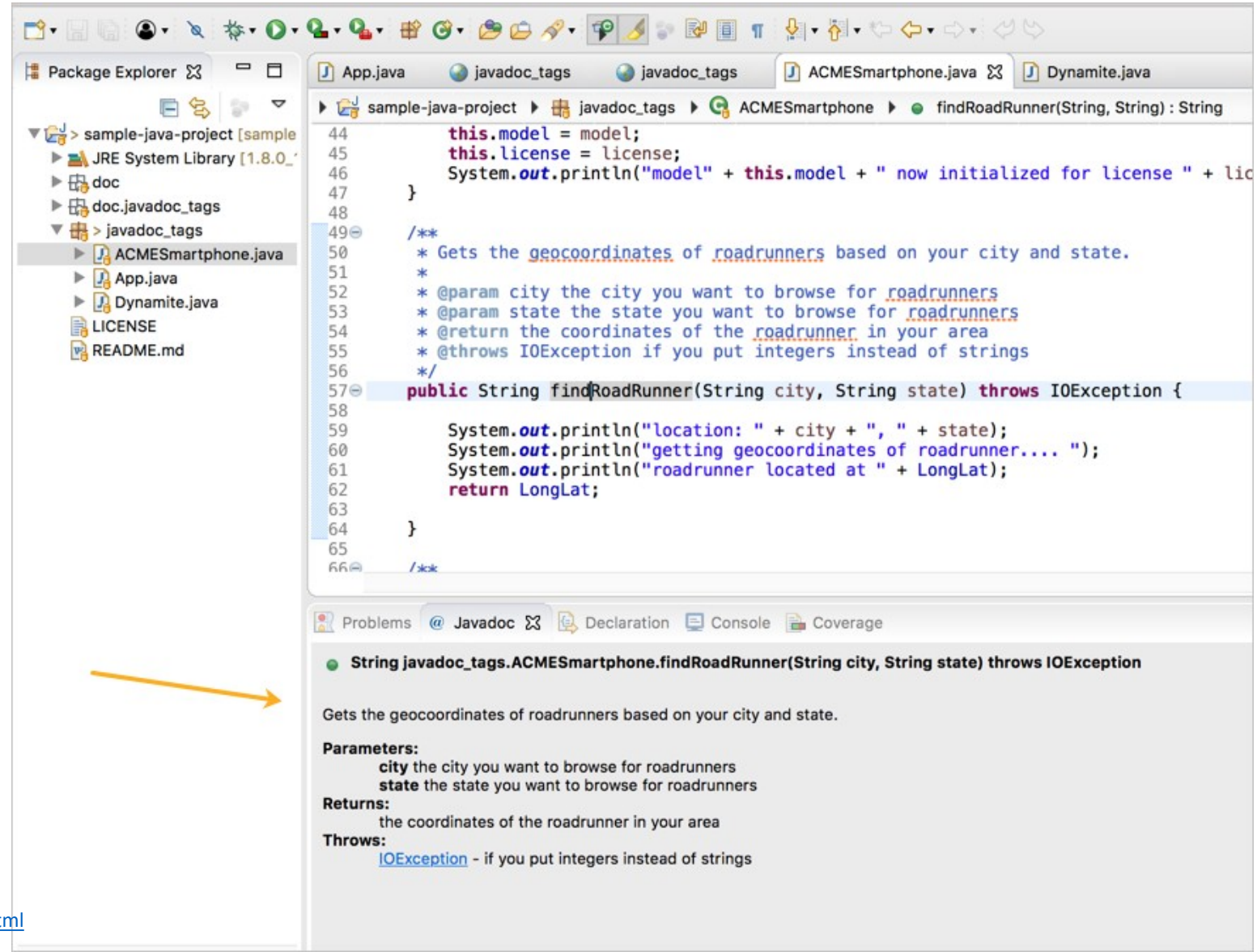
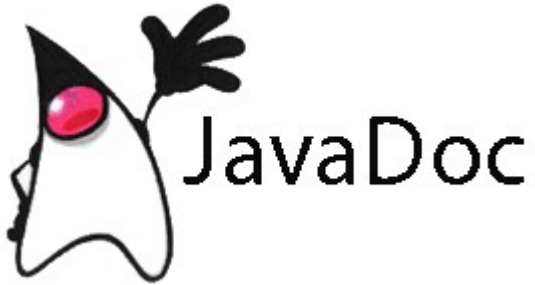
Level 4
Code

Technical documentation

The screenshot shows an IDE with the following components:

- Package Explorer:** Displays the project structure for `calimero-tools`. The `src` directory contains a package `tuwien.auto.calimero.tools` with various Java files. `Main.java` is selected.
- Code Editor:** Shows the `Main.java` file. The `usage()` method is visible, which uses `StringBuilder` to format command-line help text. Line 182 shows `sb.append(cmds[i][0]).append(" - ").append(cmds[i][1]).append("\n");`.
- Documentation Pop-up:** A yellow box provides details for the `append` method:
 - Text:** "Appends the string representation of the boolean argument to the sequence. The overall effect is exactly as if the argument were converted to a string by the method `String.valueOf(boolean)`, and the characters of that string were then appended to this character sequence."
 - Overrides:** `append(...)` in `AbstractStringBuilder`
 - Parameters:** `b` a `boolean`.
 - Returns:** a reference to this object.
- Method List:** A scrollable list of `append` overloads for `StringBuilder`, including:
 - `append(boolean b) : StringBuilder - StringBuilder`
 - `append(char c) : StringBuilder - StringBuilder`
 - `append(char[] str) : StringBuilder - StringBuilder`
 - `append(CharSequence s) : StringBuilder - StringBuilder`
 - `append(double d) : StringBuilder - StringBuilder`
 - `append(float f) : StringBuilder - StringBuilder`
 - `append(int i) : StringBuilder - StringBuilder`
 - `append(long lng) : StringBuilder - StringBuilder`
 - `append(Object obj) : StringBuilder - StringBuilder`
 - `append(String str) : StringBuilder - StringBuilder`
 - `append(StringBuffer sb) : StringBuilder - StringBuilder`
 - `append(char[] str, int offset, int len) : StringBuilder - StringBuilder`
 - `append(CharSequence s, int start, int end) : StringBuilder - StringBuilder`
 - `appendCodePoint(int codePoint) : StringBuilder - StringBuilder`

Technical documentation



The screenshot shows an IDE with a Package Explorer on the left and a code editor on the right. The Package Explorer shows a project named 'sample-java-project' with a 'javadoc_tags' folder. The code editor shows the 'ACMESmartphone.java' file with the following code:

```
44     this.model = model;
45     this.license = license;
46     System.out.println("model" + this.model + " now initialized for license " + lic
47 }
48
49 /**
50  * Gets the geocoordinates of roadrunners based on your city and state.
51  *
52  * @param city the city you want to browse for roadrunners
53  * @param state the state you want to browse for roadrunners
54  * @return the coordinates of the roadrunner in your area
55  * @throws IOException if you put integers instead of strings
56  */
57 public String findRoadRunner(String city, String state) throws IOException {
58
59     System.out.println("location: " + city + ", " + state);
60     System.out.println("getting geocoordinates of roadrunner... ");
61     System.out.println("roadrunner located at " + LongLat);
62     return LongLat;
63 }
64
65
66 /**
```

An orange arrow points from the 'findRoadRunner' method in the code editor to the Javadoc documentation in the bottom panel. The bottom panel shows the following documentation:

String javadoc_tags.ACMESmartphone.findRoadRunner(String city, String state) throws IOException

Gets the geocoordinates of roadrunners based on your city and state.

Parameters:

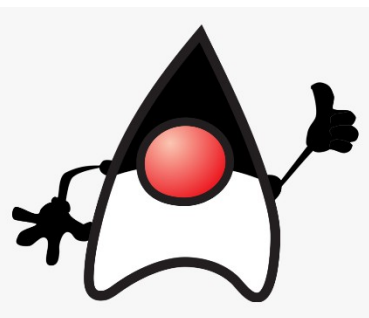
- city** the city you want to browse for roadrunners
- state** the state you want to browse for roadrunners

Returns:

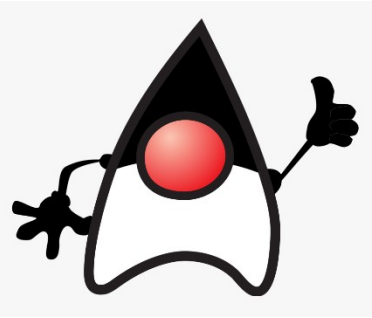
- the coordinates of the roadrunner in your area

Throws:

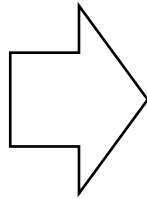
- [IOException](#) - if you put integers instead of strings



Tag	Description	Syntax
@author	Adds the author of a class.	@author name-text
{@code}	Displays text in code font without interpreting the text as HTML markup or nested javadoc tags.	{@code text}
{@docRoot}	Represents the relative path to the generated document's root directory from any generated page.	{@docRoot}
@deprecated	Adds a comment indicating that this API should no longer be used.	@deprecated deprecatedtext
@exception	Adds a Throws subheading to the generated documentation, with the classname and description text.	@exception class-name description
{@inheritDoc}	Inherits a comment from the nearest inheritable class or implementable interface.	Inherits a comment from the immediate superclass.
{@link}	Inserts an in-line link with the visible text label that points to the documentation for the specified package, class, or member name of a referenced class.	{@link package.class#member label}
{@linkplain}	Identical to {@link}, except the link's label is displayed in plain text than code font.	{@linkplain package.class#member label}
@param	Adds a parameter with the specified parameter-name followed by the specified description to the "Parameters" section.	@param parameter-name description
@return	Adds a "Returns" section with the description text.	@return description
@see	Adds a "See Also" heading with a link or text entry that points to reference.	@see reference
@serial	Used in the doc comment for a default serializable field.	@serial field-description include exclude
@serialData	Documents the data written by the writeObject() or writeExternal() methods.	@serialData data-description
@serialField	Documents an ObjectOutputStream component.	@serialField field-name field-type field-description
@since	Adds a "Since" heading with the specified since-text to the generated documentation.	@since release
@throws	The @throws and @exception tags are synonyms.	@throws class-name description
{@value}	When {@value} is used in the doc comment of a static field, it displays the value of that constant.	{@value package.class#field}
@version	Adds a "Version" subheading with the specified version-text to the generated docs when the -version option is used.	@version version-text



- generate HTML pages
- generate javadoc jars



StringBuilder (Java Platform SE 7 x +)

https://docs.oracle.com/javase/7/docs/api/

Java™ Platform Standard Ed. 7

All Classes

Packages

java.applet
java.awt
java.awt.color
java.awt.datatransfer
java.awt.dnd
java.awt.event

StatementEventListener
StAXResult
StAXSource
Streamable
StreamableValue
StreamCorruptedException
StreamFilter
StreamHandler
StreamPrintService
StreamPrintServiceFactory
StreamReaderDelegate
StreamResult
StreamSource
StreamTokenizer
StrictMath
String
StringBuffer
StringBufferInputStream
StringBuilder
StringCharacterIterator
StringContent
StringHolder
StringIndexOutOfBoundsException
StringMonitor
StringMonitorMBean
StringNameHelper
StringReader
StringRefAddr
StringSelection
StringSeqHelper
StringSeqHolder
StringTokenizer
StringValueExp

Returns:
a reference to this object.

append

public `StringBuilder` append(`String` str)

Appends the specified string to this character sequence.

The characters of the `String` argument are appended, in order, increasing the length of this sequence by the length of the argument. If str is null, then the four characters "null" are appended.

Let n be the length of this character sequence just prior to execution of the append method. Then the character at index k in the new character sequence is equal to the character at index k in the old character sequence, if k is less than n ; otherwise, it is equal to the character at index $k-n$ in the argument str.

Parameters:
str - a string.

Returns:
a reference to this object.

append

public `StringBuilder` append(`StringBuffer` sb)

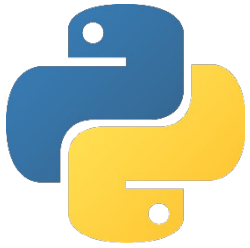
Appends the specified `StringBuffer` to this sequence.

The characters of the `StringBuffer` argument are appended, in order, to this sequence, increasing the length of this sequence by the length of the argument. If sb is null, then the four characters "null" are appended to this sequence.

Let n be the length of this character sequence just prior to execution of the append method. Then the character at index k in the new character sequence is equal to the character at index k in the old character sequence, if k is less than n ; otherwise, it is equal to the character at index $k-n$ in the argument sb.

Parameters:
sb - the `StringBuffer` to append.

Returns:

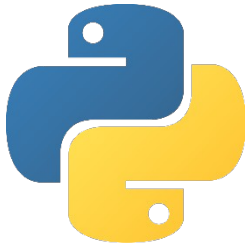


Python docstrings

```
class DocsBuilder:
    """
    Class used for automatically generating HTML-based documentation from
    Python code. Note that function docstrings must also follow NumPy/SciPy
    docstring formatting conventions.
    """
    def __init__(self, docs_dir='docs', offline=False):
        """
        ... docstring ...
        """
        ... code ...

    def extract(self, code):
        """
        ... docstring ...
        """
        ... code ...

    def output(code, filename, path="docs"):
```



Python doc conventions

```
"""
This is a javadoc style.

@param param1: this is a first param
@param param2: this is a second param
@return: this is a description of what is returned
@raise KeyError: raises an exception
"""
```

Epytext

```
"""
This is an example of Google style.

Args:
    param1: This is the first param.
    param2: This is a second param.

Returns:
    This is a description of what is returned.

Raises:
    KeyError: Raises an exception.
"""
```

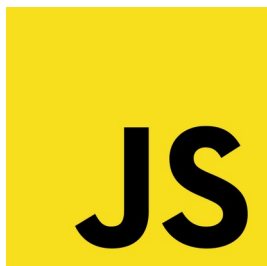
Google

```
"""
This is a reST style.

:param param1: this is a first param
:param param2: this is a second param
:returns: this is a description of what is returned
:raises KeyError: raises an exception
"""
```

recommended

reStructuredText -> used by Sphinx



JavaScript documentation: JSDoc

Initial release	1999; 22 years ago
Latest release	3.6.3 (15 July 2019; 2 years ago)
Type of format	Programming documentation Format
Contained by	JavaScript source files
Extended from	JavaDoc
Open format?	Yes
Website	jsdoc.app

```
/** @class Circle representing a circle. */
class Circle {
  /**
   * Creates an instance of Circle.
   *
   * @author: moi
   * @param {number} r The desired radius of the circle.
   */
  constructor(r) {
    /** @private */ this.radius = r
    /** @private */ this.circumference = 2 * Math.PI * r
  }

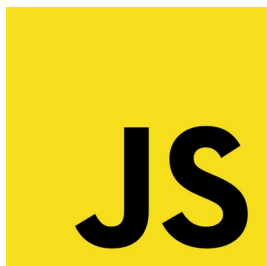
  /**
   * Creates a new Circle from a diameter.
   *
   * @param {number} d The desired diameter of the circle.
   * @return {Circle} The new Circle object.
   */
  static fromDiameter(d) {
    return new Circle(d / 2)
  }

  /**
   * Calculates the circumference of the Circle.
   *
   * @deprecated since 1.1.0; use getCircumference instead
   * @return {number} The circumference of the circle.
   */
  calculateCircumference() {
    return 2 * Math.PI * this.radius
  }
}
```

```
/**
 * Returns the pre-computed circumference of the Circle.
 *
 * @return {number} The circumference of the circle.
 * @since 1.1.0
 */
getCircumference() {
  return this.circumference
}

/**
 * Find a String representation of the Circle.
 *
 * @override
 * @return {string} Human-readable representation of this Circle.
 */
toString() {
  return `[A Circle object with radius of ${this.radius}.]`
}

/**
 * Prints a circle.
 *
 * @param {Circle} circle
 */
function printCircle(circle) {
  /** @this {Circle} */
  function bound() { console.log(this) }
  bound.apply(circle)
}
```



JavaScript documentation: JSDoc

Initial release	1999; 22 years ago
Latest release	3.6.3 (15 July 2019; 2 years ago)
Type of format	Programming documentation Format
Contained by	JavaScript source files
Extended from	JavaDoc
Open format?	Yes
Website	jsdoc.app

Tag	Description
<code>@author</code>	Developer's name
<code>@constructor</code>	Marks a function as a constructor
<code>@deprecated</code>	Marks a method as deprecated
<code>@exception</code>	Synonym for <code>@throws</code>
<code>@exports</code>	Identifies a member that is exported by the module
<code>@param</code>	Documents a method parameter; a datatype indicator can be added between curly braces
<code>@private</code>	Signifies that a member is private
<code>@returns</code>	Documents a return value
<code>@return</code>	Synonym for <code>@returns</code>
<code>@see</code>	Documents an association to another object
<code>@todo</code>	Documents something that is missing/open
<code>@this</code>	Specifies the type of the object to which the keyword <code>this</code> refers within a function.
<code>@throws</code>	Documents an exception thrown by a method
<code>@version</code>	Provides the version number of a library

Self-documenting code

- Make source code easier to read and understand
- Minimize the effort required to maintain or extend legacy systems
- Reduce the need for users and developers of a system to consult secondary documentation sources such as code comments or software manuals
- Facilitate automation through self-contained knowledge representation

Self-documenting code

- Make source code easier to read and understand
- Minimize the effort required to maintain or extend legacy systems
- Reduce the need for users and developers of a system to consult secondary documentation sources such as code comments or software manuals
- Facilitate automation through self-contained knowledge representation

```
size_t count_alphabetic_chars(const char *text)
{
    if (text == NULL)
        return 0;

    size_t count = 0;

    while (*text != '\0')
    {
        if (is_alphabetic(*text))
            count++;
        text++;
    }

    return count;
}
```

Self-documenting code

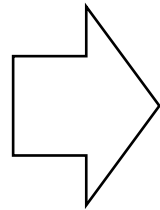
- Make source code easier to read and understand
- Minimize the effort required to maintain or extend legacy systems
- Reduce the need for users and developers of a system to consult secondary documentation sources such as code comments or software manuals
- Facilitate automation through self-contained knowledge representation



Self-documenting code

1. Move code to function

```
var width = (value - 0.5) * 16;
```



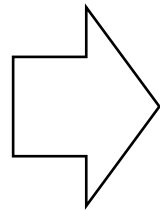
```
var width = emToPixels(value);  
  
function emToPixels(ems) {  
    return (ems - 0.5) * 16;  
}
```

Self-documenting code

2. Expression is replaced with a variable

```
var isVisible = el.offsetWidth && el.offsetHeight;  
if(!isVisible) {  
}
```

```
return a * b + (c / d);
```

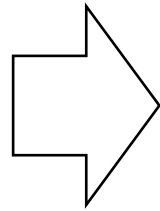


```
var multiplier = a * b;  
var divisor = c / d;  
return multiplier + divisor;
```

Self-documenting code

3. Class and module interfaces

```
class Box {  
    setState(state) {  
        this.state = state;  
    }  
  
    getState() {  
        return this.state;  
    }  
}
```



```
class Box {  
    open() {  
        this.state = 'open';  
    }  
  
    close() {  
        this.state = 'closed';  
    }  
  
    isOpen() {  
        return this.state === 'open';  
    }  
}
```

Self-documenting code

4. Group your code

```
var foo = 1;  
  
blah()  
xyz();  
  
bar(foo);  
baz(1337);  
quux(foo);
```

Self-documenting code

5. Give another name to the function

Omit to use obscure words:

```
handleButtons(), manageLinks()
```

Use active verbs like "send":

```
cutLawn(), sendFolder()
```

Self-documenting code

6. Give other names to variables

Do not use silly variable names



```
a, b, c, variable1, other, ...
```

7. Follow the same naming conventions

```
var element = getElement();
```



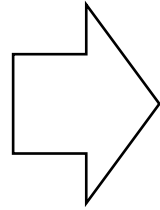
```
var node = getElement();
```

Self-documenting code

8. Avoid utilizing syntax tricks



```
imStrange && doMagic();
```



```
if(imTricky) {  
    doMagic();  
}
```

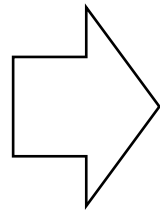
Self-documenting code

9. Use named constants

```
const BUY_HAPPINESS = 42;
```

10. Omit boolean flags

```
myStuff.setData({ x: 1 }, true);
```

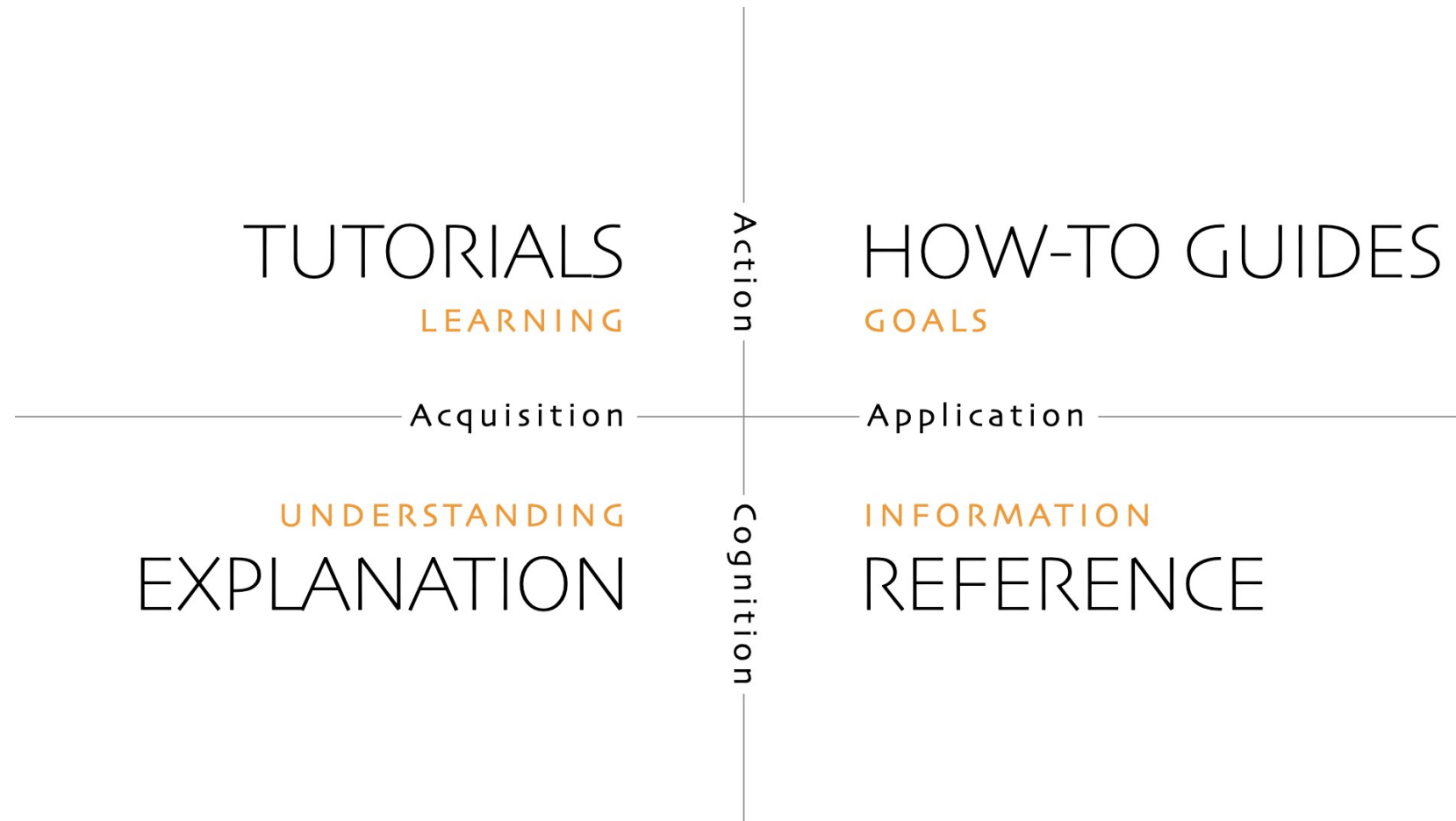


```
myStuff.mergeData({ x: 1 });
```

Documentation for the end user

Diátaxis

A systematic approach to technical documentation authoring.



Documentation for the end user

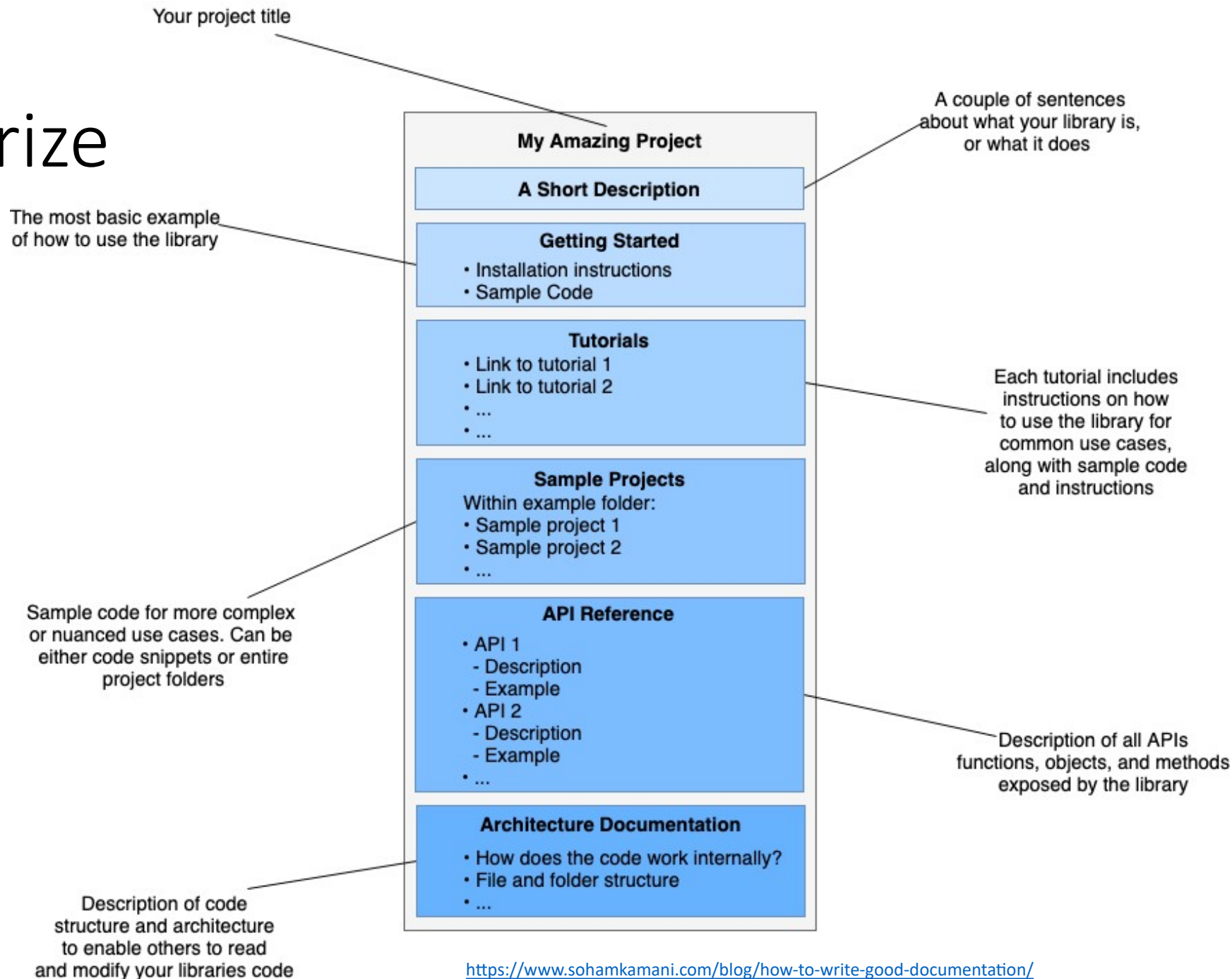
Diátaxis

A systematic approach to technical documentation authoring.

Describes how the program is installed and used

- **Tutorials:** *A tutorial is an **experience** that takes place under the guidance of a tutor. A tutorial is always **learning-oriented**.*
- **How-to guides:** *How-to guides are **directions** that guide the reader through a problem or towards a result. How-to guides are **goal-oriented**.*
- **References:** *Reference guides are **technical descriptions** of the machinery and how to operate it. Reference material is **information-oriented**.*
- **Explanations:** *Explanation is a discursive treatment of a subject, that permits reflection. Explanation is **understanding-oriented**.*

To summarize



Case study 1

 pandas

Getting started | User Guide | API reference | Development | Release notes

Search the docs ...

pandas documentation

Date: Dec 12, 2021 **Version:** 1.3.5

Download documentation: [PDF Version](#) | [Zipped HTML](#)

Previous versions: Documentation of previous pandas versions is available at pandas.pydata.org.

Useful links: [Binary Installers](#) | [Source Repository](#) | [Issues & Ideas](#) | [Q&A Support](#) | [Mailing List](#)

pandas is an open source, BSD-licensed library providing high-performance, easy-to-use data structures and data analysis tools for the Python programming language.



Getting started

New to *pandas*? Check out the getting started guides. They contain an introduction to *pandas*' main concepts and links to additional tutorials.

[To the getting started guides](#)



User guide

The user guide provides in-depth information on the key concepts of pandas with useful background information and explanation.

[To the user guide](#)



API reference

The reference guide contains a detailed description of the pandas API. The reference describes how the methods work and which parameters can be used. It assumes that you have an understanding of the key concepts.

[To the reference guide](#)



Developer guide

Saw a typo in the documentation? Want to improve existing functionalities? The contributing guidelines will guide you through the process of improving pandas.

[To the development guide](#)

Case study 2

ESP-IDF Programming Guide

 ESPRESSIF

ESP32

master (latest)

Search docs

Get Started

API Reference

H/W Reference

API Guides

ESP-IDF 5.0 Migration Guides

Libraries and Frameworks

Contribute

Versions

Resources

Copyrights

About

语言/Languages

<https://docs.espressif.com/projects/esp-idf/>

» ESP-IDF Programming Guide

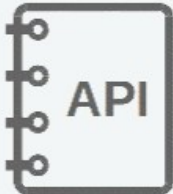
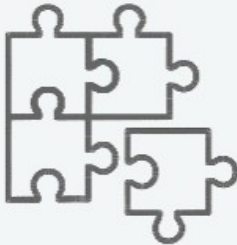
Edit on GitHub

ESP-IDF Programming Guide

[中文]

This is the documentation for Espressif IoT Development Framework ([esp-idf](#)). ESP-IDF is the official development framework for the [ESP32](#), [ESP32-S](#) and [ESP32-C Series SoCs](#).

This document describes using ESP-IDF with the ESP32 SoC. To switch to a different SoC target, choose target from the dropdown in the upper left.

		
Get Started	API Reference	H/W Reference
		
API Guides	Contribute	Resources

• Index

The README file

- A description of what the project is for.
 - What is this repo or project? (You can reuse the repo description you used earlier because this section doesn't have to be long.)
 - How does it work?
 - Who will use this repo or project?
 - What is the goal of this project?
- Instructions for how to develop, use, and test the code.
- Instructions for how people can help.
- List the licensing information for your project.
- List the contact information for your team as well as where to ask questions.

see <https://github.com/18F/open-source-guide/blob/18f-pages/pages/making-readmes-readable.md>

To read for MCQ test

- The Diátaxis systematic approach to creating better documentation:
 - [Tutorials](#)
 - [How-to guides](#)
 - [Reference](#)
 - [Explanation](#)

Technological foundations of software development

Document, license, publish, maintain your software

Part 2: Software licenses

ICM – Computer Science Major – Course unit on Technological foundations of computer science

M1 Cyber Physical and Social Systems – Course unit on CPS2 engineering and development, Part 2: Technological foundations of software development

Maxime Lefrançois <https://maxime-lefrancois.info>

online: <https://ci.mines-stetienne.fr/cps2/course/tfsd/>

Creative Commons

- Set of licenses for published works

The spectrum of rights



CREATIVE COMMONS LICENSES

	PUBLIC DOMAIN
	CC BY
	CC BY-SA
	CC BY-ND
	CC BY-NC
	CC BY-NC-SA
	CC BY-NC-ND



COPY
& PUBLISH



ATTRIBUTION
REQUIRED



COMMERCIAL
USE



MODIFY
& ADAPT



CHANGE
LICENSE



You can redistribute
(copy, publish, display,
communicate, etc.)



You have to attribute
the original work



You can use the work
commercially



You can modify and
adapt the original work



You can choose license
type for your adaptations
of the work.

The Creative Commons licences explained.

Software license

Contract by which the owner of the copyright on a computer program defines with his co-contractor (operator or user) the conditions under which this program can be used, distributed or modified.

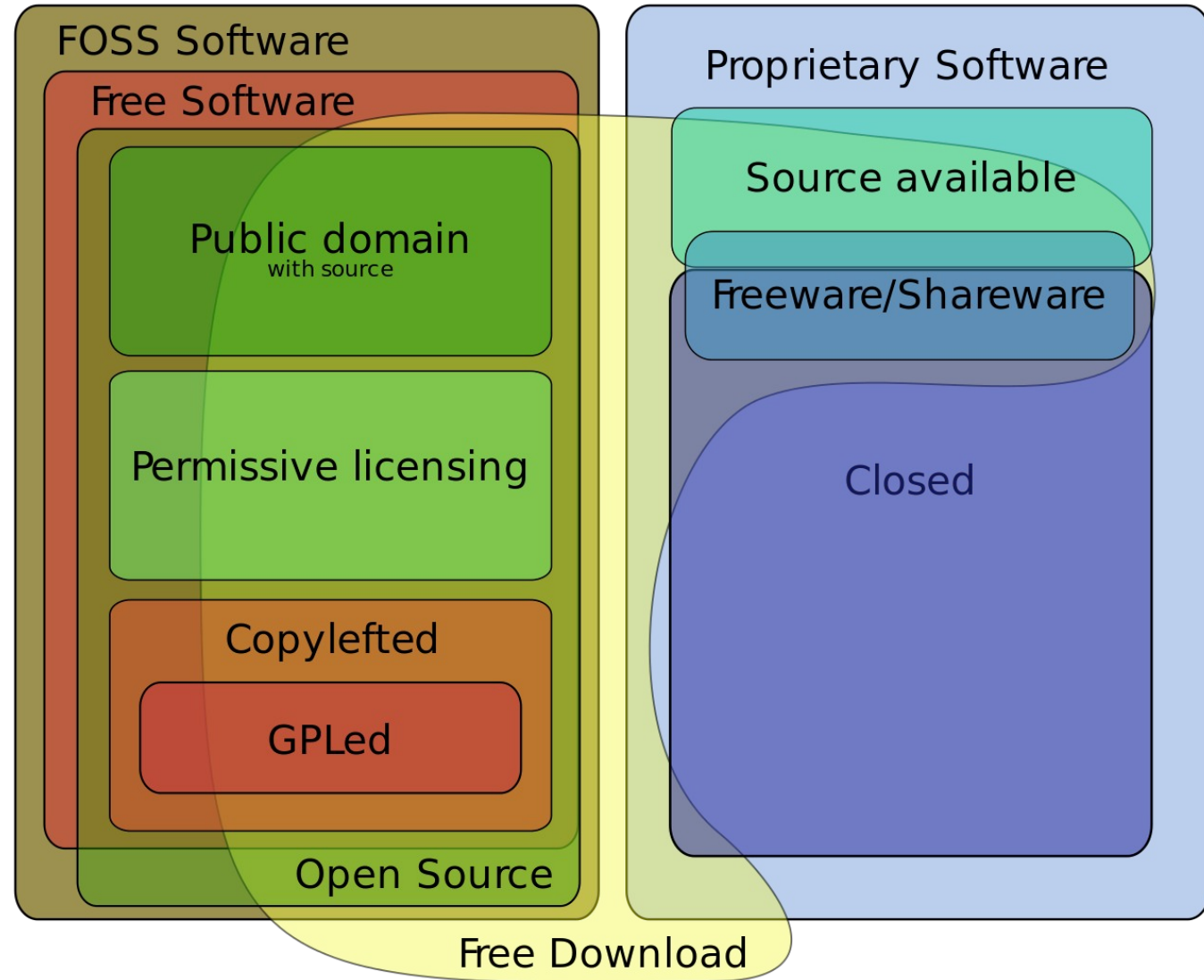
Software license

Software licenses and rights granted in context of the copyright according to [Mark Webbink](#).^[2] Expanded by freeware and sublicensing.

Rights granted	Public domain	Permissive FOSS license (e.g. BSD license)	Copyleft FOSS license (e.g. GPL)	Freeware/Shareware/Freemium	Proprietary license	Trade secret
Copyright retained	No	Yes	Yes	Yes	Yes	Very strict
Right to perform	Yes	Yes	Yes	Yes	Yes	No
Right to display	Yes	Yes	Yes	Yes	Yes	No
Right to copy	Yes	Yes	Yes	Often	No	Lawsuits are filed by the owner against copyright infringement the most
Right to modify	Yes	Yes	Yes	No	No	No
Right to distribute	Yes	Yes, under same license	Yes, under same license	Often	No	No
Right to sublicense	Yes	Yes	No	No	No	No
Example software	SQLite, ImageJ	Apache web server, ToyBox	Linux kernel, GIMP, OBS	Irfanview, Winamp, <i>League of Legends</i>	Windows, the majority of commercial video games and their DRMs, Spotify, xSplit, TIDAL	Server-side Cloud computing programs and services, forensic applications, and other line-of-business work.

Free software foundation

Free Software Foundation	
	FREE SOFTWARE FOUNDATION
Abbreviation	FSF
Formation	October 4, 1985; 36 years ago ^[1]
Founder	Richard Stallman (GNU project)
Type	501(c)(3) non-profit organization
Legal status	501(c)(3)
Purpose	Educational
Headquarters	Boston, Massachusetts, US
Region served	Worldwide
Membership	Individuals
President	Geoffrey Knauth
Budget	\$1,373,645 in FY 2017 ^[2]
Staff	14 ^[3]
Website	www.fsf.org



Open Source licenses



Open Source Initiative

Guaranteeing the 'our' in source...

Open source licenses are licenses that comply with the Open Source Definition <https://opensource.org/osd>

— in brief, they allow software to be freely used, modified, and shared.

Apache License 2.0

BSD 3-Clause "New" or "Revised" license

BSD 2-Clause "Simplified" or "FreeBSD" license

GNU General Public License (GPL)

GNU Library or "Lesser" General Public License (LGPL)

MIT license

Mozilla Public License 2.0

Common Development and Distribution License

Eclipse Public License version 2.0

...

FOSS Licenses (free and open-source software)

							
Type	Permissive	Permissive	Permissive	Permissive	Copyleft	Copyleft	Copyleft
Provides copyright protection	✓ TRUE	✓ TRUE	✓ TRUE	✓ TRUE	✓ TRUE	✓ TRUE	✓ TRUE
Can be used in commercial applications	✓ TRUE	✓ TRUE	✓ TRUE	✓ TRUE	✓ TRUE	✓ TRUE	✓ TRUE
Provides an explicit patent license	✓ TRUE	✗ FALSE	✗ FALSE	✗ FALSE	✗ FALSE	✗ FALSE	✗ FALSE
Can be used in proprietary (closed source) projects	✓ TRUE	✓ TRUE	✓ TRUE	✓ TRUE	✗ FALSE	✗ FALSE partially	✗ FALSE for web
Popular open-source and free projects	Kubernetes Swift Firebase	Django React Flutter	Angular.js JQuery, .NET Core Laravel	Joomla Notepad++ MySQL	Qt SharpDevelop	SugarCRM Launchpad	

Choose an open source license

An open source license protects contributors and users. Businesses and savvy developers won't touch a project without this protection.

{ Which of the following best describes your situation? }



I need to work in a community.

Use the **license preferred by the community** you're contributing to or depending on. Your project will fit right in.

If you have a dependency that doesn't have a license, ask its maintainers to **add a license**.



I want it simple and permissive.

The **MIT License** is short and to the point. It lets people do almost anything they want with your project, like making and distributing closed source versions.

Babel, **.NET Core**, and **Rails** use the MIT License.



I care about sharing improvements.

The **GNU GPLv3** also lets people do almost anything they want with your project, *except* distributing closed source versions.

Ansible, **Bash**, and **GIMP** use the GNU GPLv3.

Display the license in your Git repository

Determining the location of your license

Most people place their license text in a file named `LICENSE.txt` (or `LICENSE.md` or `LICENSE.rst`) in the root of the repository; [here's an example from Hubot](#).

Some projects include information about their license in their README. For example, a project's README may include a note saying "This project is licensed under the terms of the MIT license."

As a best practice, we encourage you to include the license file with your project.

Applying a license to a repository with an existing license

The license picker is only available when you create a new project on GitHub. You can manually add a license using the browser. For more information on adding a license to a repository, see "[Adding a license to a repository](#)."

☐ **Initialize this repository with a README**
This will allow you to `git clone` the repository immediately.

Add .gitignore: **None** ▼

Add a license: **None** ▼



Technological foundations of software development

Document, license, publish, maintain your software

Part 3: Publish your software

ICM – Computer Science Major – Course unit on Technological foundations of computer science

M1 Cyber Physical and Social Systems – Course unit on CPS2 engineering and development, Part 2: Technological foundations of software development

Maxime Lefrançois <https://maxime-lefrancois.info>

online: <https://ci.mines-stetienne.fr/cps2/course/tfsd/>

Publish your software

Software publishing is a term that describes the overall process of designing and distributing a software package or collection of software packages.

Package repositories

☰ The Central Repository



```
<distributionManagement>
  <snapshotRepository>
    <id>ossrh</id>
    <url>https://s01.oss.sonatype.org/content/repositories/snapshots</url>
  </snapshotRepository>
</distributionManagement>
<build>
  <plugins>
    <plugin>
      <groupId>org.sonatype.plugins</groupId>
      <artifactId>nexus-staging-maven-plugin</artifactId>
      <version>1.6.7</version>
      <extensions>true</extensions>
      <configuration>
        <serverId>ossrh</serverId>
        <nexusUrl>https://s01.oss.sonatype.org/</nexusUrl>
        <autoReleaseAfterClose>true</autoReleaseAfterClose>
      </configuration>
    </plugin>
    ...
  </plugins>
</build>
```

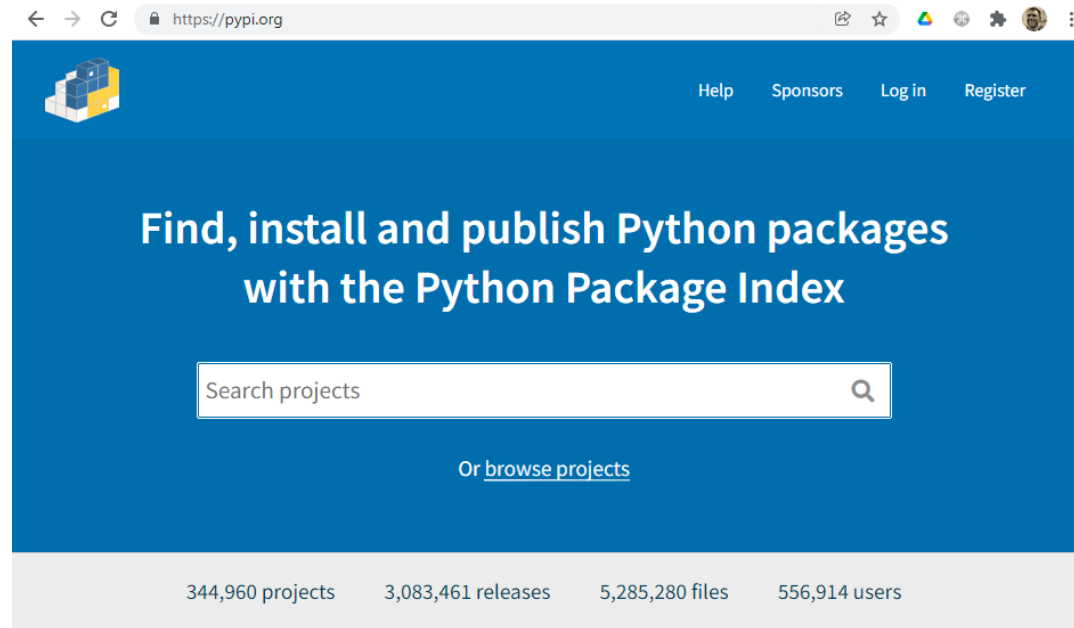
example with Maven: in the pom.xml

- Guide to start publishing your code:
<https://central.sonatype.org/publish/publish-guide/>
 - sign up
 - generate artifacts: binaries, source, javadoc
 - sign your artifacts (GPG)
 - publish : > mvn clean deploy nexus-staging:release

```
<settings>
  <servers>
    <server>
      <id>ossrh</id>
      <username>your-jira-id</username>
      <password>your-jira-pwd</password>
    </server>
  </servers>
</settings>
```

example with Maven: in the ~/.m2/settings.xml

Package repositories



- Guide to start publishing your code:
<https://packaging.python.org/en/latest/tutorials/packaging-projects/>

Publish on github/gitlab

- <https://docs.github.com/en/repositories/releasing-projects-on-github/managing-releases-in-a-repository>

3 months ago

atom-build

v1.58.0

550c1da

Compare

1.58.0

Latest

Notable Changes

- #22315 - Update to macOS Big Sur icon.
- #22424 - Fix reopening a project in safeMode and devMode.
- #22123 - Improve contrast on Windows app icons
- atom/archive-view#73 - Add ability to collapse archived directories (zip, tar, e.t.c)
- atom/bracket-matcher#405 - Handle multicursor selection inside brackets
- atom/find-and-replace#932 - Add "Open in New Tab" and "Open in New Window" right-click context menu options

Thanks to @octocat and @hubot for their contributions to this release.

► All Changes

Contributors



octocat and hubot

► Assets 18

👍 39 🙏 7 🌟 16 ❤️ 11 🍷 7 👁 9 54 people reacted

Researchers: get a DOI with Zenodo

- Digital Object Identifiers (DOIs)
- For academics, you can now publish your software (or data) and obtain a permanent identifier so that it can be cited in scientific publications

The screenshot shows the Zenodo interface for a software release. At the top is a blue header with the Zenodo logo, a search bar, and links for 'Upload' and 'Communities'. On the right are 'Log in' and 'Sign up' buttons. Below the header is a yellow banner stating 'There is a newer version of this record available.' The main content area shows the record title 'sparql-generate/sparql-generate: 2.0.6' with a date of 'April 15, 2021' and tags for 'Software' and 'Open Access'. The authors listed are Maxime Lefrançois, Bakerrally Noorani, KHALFI el mehdi, Omar Alqawasmeh, Omar Alqawasmeh, Antoine Zimmermann, Maksim Kolchin, and Miguel Ceriani. The description is 'SPARQL-Generate implementation over Apache Jena'. To the right of the title, statistics show 58 views and 14 downloads. Below the title is a 'Preview' section showing a file tree for 'sparql-generate-sparql-generate-41e0dd9'. The tree includes a '.github' directory with 'workflows' (maven.yml, release.yml, test.yml, website.yml), '.gitignore', 'LICENSE', 'README.md', 'release.sh', 'snapshot.sh', and a 'sparql-generate-all' directory with its own '.gitignore', 'README.md', and 'pom.xml'. File sizes are listed next to each item. At the bottom, a table lists the files, showing 'sparql-generate/sparql-generate-2.0.6.zip' with a size of 26.3 MB and options to 'Preview' or 'Download'. The MD5 hash is also provided. On the right side of the page, there are two boxes: 'Available in' showing 'GitHub' and 'Indexed in' showing 'OpenAIRE'. At the bottom right, a 'Publication date' box shows 'April 15, 2021', a 'DOI' box shows '10.5281/zenodo.4692775', a 'Related identifiers' box shows a link to the GitHub repository, and a 'License (for files)' box shows 'Other (Open)'.

zenodo Search Upload Communities Log in Sign up

There is a **newer version** of this record available.

April 15, 2021 Software Open Access

sparql-generate/sparql-generate: 2.0.6

Maxime Lefrançois; Bakerrally Noorani; KHALFI el mehdi; Omar Alqawasmeh; Omar Alqawasmeh; Antoine Zimmermann; Maksim Kolchin; Miguel Ceriani

SPARQL-Generate implementation over Apache Jena

Preview

sparql-generate-2.0.6.zip

The previewer is not showing all the files

sparql-generate-sparql-generate-41e0dd9

- .github
 - workflows
 - maven.yml 942 Bytes
 - release.yml 3.5 kB
 - test.yml 711 Bytes
 - website.yml 802 Bytes
 - .gitignore 535 Bytes
 - LICENSE 573 Bytes
 - README.md 151 Bytes
 - release.sh 1.3 kB
 - snapshot.sh 725 Bytes
 - sparql-generate-all
 - .gitignore 497 Bytes
 - README.md 2.2 kB
 - pom.xml 3.2 kB

Files (26.3 MB)

Name	Size	
sparql-generate/sparql-generate-2.0.6.zip	26.3 MB	Preview Download

md5:13417b8699d7bb8b72260cb4eac05e2e

58 views 14 downloads See more details...

Available in

GitHub

Indexed in

OpenAIRE

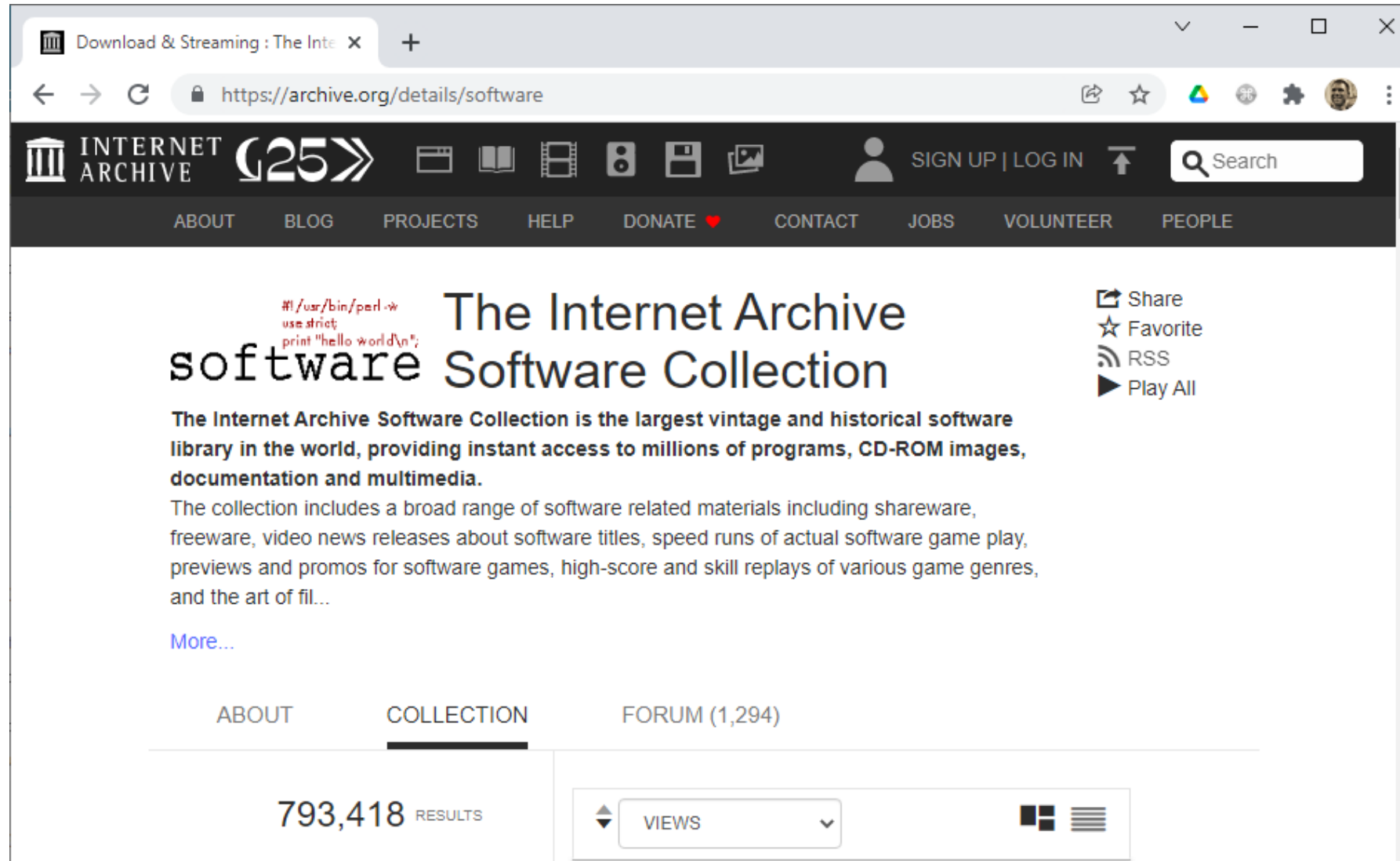
Publication date: April 15, 2021

DOI: DOI 10.5281/zenodo.4692775

Related identifiers: Supplement to https://github.com/sparql-generate/sparql-generate/tree/2.0.6

License (for files): Other (Open)

Software archives



The screenshot shows a web browser window with the address bar displaying `https://archive.org/details/software`. The page header includes the Internet Archive logo, a navigation menu with links like ABOUT, BLOG, PROJECTS, HELP, DONATE, CONTACT, JOBS, VOLUNTEER, and PEOPLE, and a search bar. The main content area features the title "The Internet Archive software Software Collection" with a small code snippet: `#!/usr/bin/perl -w
use strict;
print "hello world\n";`. Below the title, a paragraph describes the collection as the largest vintage and historical software library in the world. A "More..." link is provided. On the right, there are share and favorite options. At the bottom, a navigation bar shows "ABOUT", "COLLECTION" (selected), and "FORUM (1,294)". The results section displays "793,418 RESULTS" and a "VIEWS" dropdown menu.

Download & Streaming : The Inte x +

← → ↻ 🔒 `https://archive.org/details/software` 📄 ☆ 🌐 ⚙️ 👤 ⋮

INTERNET ARCHIVE 25 >> 🖥️ 📖 🎬 🎧 💾 🖼️ 👤 SIGN UP | LOG IN 🔍 Search

ABOUT BLOG PROJECTS HELP DONATE ❤️ CONTACT JOBS VOLUNTEER PEOPLE

`#!/usr/bin/perl -w
use strict;
print "hello world\n";` The Internet Archive
software Software Collection

[Share](#)
[Favorite](#)
[RSS](#)
[Play All](#)

The Internet Archive Software Collection is the largest vintage and historical software library in the world, providing instant access to millions of programs, CD-ROM images, documentation and multimedia.

The collection includes a broad range of software related materials including shareware, freeware, video news releases about software titles, speed runs of actual software game play, previews and promos for software games, high-score and skill replays of various game genres, and the art of fil...

[More...](#)

ABOUT **COLLECTION** FORUM (1,294)

793,418 RESULTS

⬆️ VIEWS ⬆️ 🖱️ ☰

Technological foundations of software development

Document, license, publish, maintain your software

Part 4: Maintaining your software

ICM – Computer Science Major – Course unit on Technological foundations of computer science

M1 Cyber Physical and Social Systems – Course unit on CPS2 engineering and development, Part 2: Technological foundations of software development

Maxime Lefrançois <https://maxime-lefrancois.info>

online: <https://ci.mines-stetienne.fr/cps2/course/tfsd/>

Building communities - GitHub Docs

Building communities

Free, Pro, & Team English Search topics, products...

Learn best practices for moderating and setting up collaborative, safe, and effective communities using GitHub's community-tested tools.

Guides	Popular	What's new View all →
Setting guidelines for repository contributors You can create guidelines to communicate how people should contribute to your...	Creating a pull request template for your repository	Table of content for Wikis August 19
Adding a code of conduct to your project Adopt a code of conduct to define community standards, signal a welcoming...	Reporting abuse or spam	
Managing disruptive comments You can hide, edit, or delete comments on issues, pull requests, and commits.	Adding a license to a repository	
	Configuring issue templates for your repository	

Maintain your software and get the most out of your users

GitHub Issues

Learn how you can use GitHub Issues to plan and track your work.

Quickstart Overview

- Issues
- Branches
- Milestones
- Project boards
- ...

Guides [View all →](#)

Creating an issue

Issues can be created in a variety of ways, so you can choose the most convenient...

Quickstart for projects (beta)

Experience the speed, flexibility, and customization of projects (beta) by...

Best practices for managing projects (beta)

Learn tips for managing your projects on GitHub.

Popular

About issues

About projects (beta)

About task lists

About issue and pull request templates

Managing labels

What's new [View all →](#)

The new GitHub Issues – 12/02 update

December 02

Improvements to the @mention suggester

November 23

The new GitHub Issues – 11/18 update

November 18

Technological foundations of software development

Document, license, publish, maintain your software

ICM – Computer Science Major – Course unit on Technological foundations of computer science

M1 Cyber Physical and Social Systems – Course unit on CPS2 engineering and development, Part 2: Technological foundations of software development

Maxime Lefrançois <https://maxime-lefrancois.info>

online: <https://ci.mines-stetienne.fr/cps2/course/tfsd/>

... Your turn

Complete the TODO section:

https://ci.mines-stetienne.fr/cps2/course/tfsd/course-6.html#_todos