31. Documentation as Synchronized Dependent Model in a Macromodel

Documentation Generation as App for RAG

Prof. Dr. rer. nat. Uwe Aßmann
Institut für Software- und Multimediatechnik
Lehrstuhl Softwaretechnologie
Fakultät für Informatik
TU Dresden
http://st.inf.tu-dresden.de
Version 21-0.3, 08.01.22

1) Tasks
2) Template-Driven Documentation Tools
3) Literate Programming
4) Elucidative Modeling and Documentation Tools
5) Web-based API Documentation Generators

References

- D. E. Knuth, Literate Programming, The Computer Journal, Volume 27, Issue 2, 1984, Pages 97–111, https://doi.org/10.1093/comjnl/27.2.97
- D. Cordes and M. Brown, "The literate-programming paradigm," in Computer, vol. 24, no. 6, pp. 52-61, June 1991, doi: 10.1109/2.86838.
- Kurt Nørmark. Elucidative programming. Nordic Journal of Computing, 2000. Citeseer: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.408.2506&rep=rep1&type=pdf
- C. Wilke, A. Bartho, J. Schroeter, S. Karol, U. Aßmann. Elucidative Development for Model-Based Documentation and Language Specification (Extended Version). Technische Universität Dresden. Institut fur Software- und Multimediatechnik. Technical Reports TUD-FI12-01-Januar 2012, ISSN 1430-211X.
  - http://nbn-resolving.de/urn:nbn:de:bsz:14-qucosa-83442
- Andreas Bartho. Elucidative Modeling. PhD thesis, Technische Universität Dresden, Fakultät Informatik, May 2014.
  - http://nbn-resolving.de/urn:nbn:de:bsz:14-qucosa-208060
  - https://www.linkedin.com/pub/andreas-bartho/ba/922/8a4?trk=pub-pbmap

Interesting

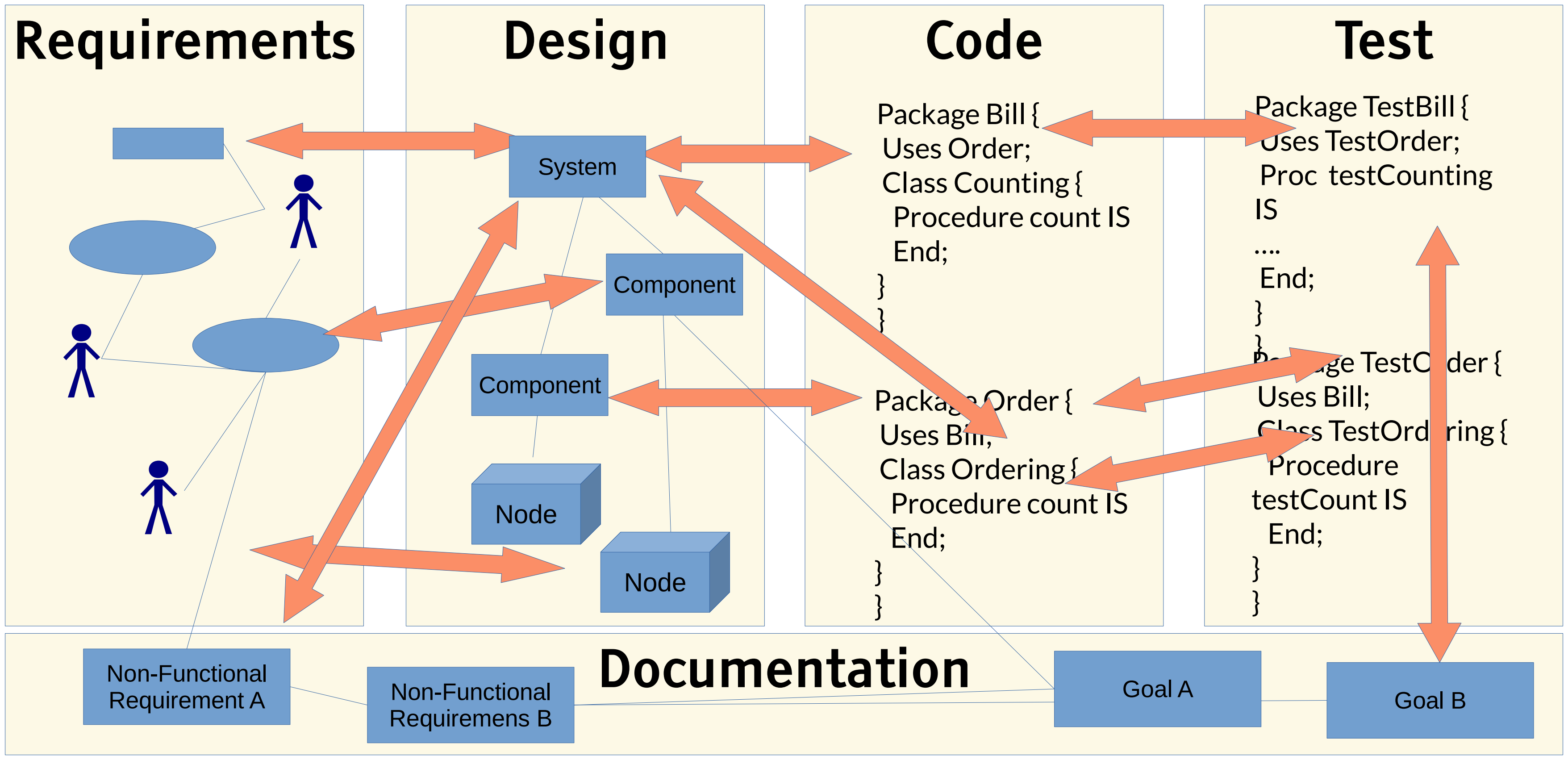
- https://www.writethedocs.org/ is a conference for documentation practicioneers
- https://waset.org/software-implementation-and-software-documentation-conference

31.1 Tasks of Documentation Tools

http://en.wikipedia.org/wiki/Software_documentation

Q12: The ReDoDeCT Problem and its Macromodel

- The **ReDoDeCT problem** is the problem how requirements, documentation, design, code, and tests are related (→ V model)
- Mappings between the Requirements model, Documentation files, Design model, Code, Test cases
- A **ReDoDeCT macromodel** has maintained mappings between all 5 models



Basics of Software Documentation

- Documentation is a means of **communication** to keep software alive
  - between developers and future developers
  - between coders and testers
  - between developers and managers (for reviews and audits)
- Problems:
  - Documentation ***ages*** because code is modified and evolved (***documentation aging)***
  - Good documentation costs time and money
- Different kinds of documentation:
  - **Generated documentation** is derived from code and models
  - **Integrated Documentation** is derived from the code (e.g., in comments), e.g., JavaDoc
  - **Elucidative Documentation**, derives both from another and keeps it consistent (generative or round-trip engineering)
- Standards:
  - national DIN 66230, 66231, 66232, 66270(1998)
  - international ISO/IEC 6592(2000), ISO/IEC 18019(2004)

Without documentation, a program is not software

Quelle: [24 S. 241 ff.]

Taxonomy of Documentation Documents

- **User documentation** (Benutzerdokumentation) explains the program to end users
  - Tutorials, user handbook, online documentation
- **System documentation** for installation, test cases, code documentation, maintenance, operations
  - **API documentation** documents interfaces of the system or framework, to let programmers use them for writing apps
  - **Architecture documentation** to highlight the architectural structure of the software, e.t., with arc42 (https://www.arc42.de/)
- **Project documentation**
  - Developer documentation
  - Project documentation (project plan, requriements specification, status reports, after study)
- **Quality documentation**
  - Test-, review, audit documentation
- **Process documentation**
  - Standards, processes

Quelle: [24 S. 245 ff.]

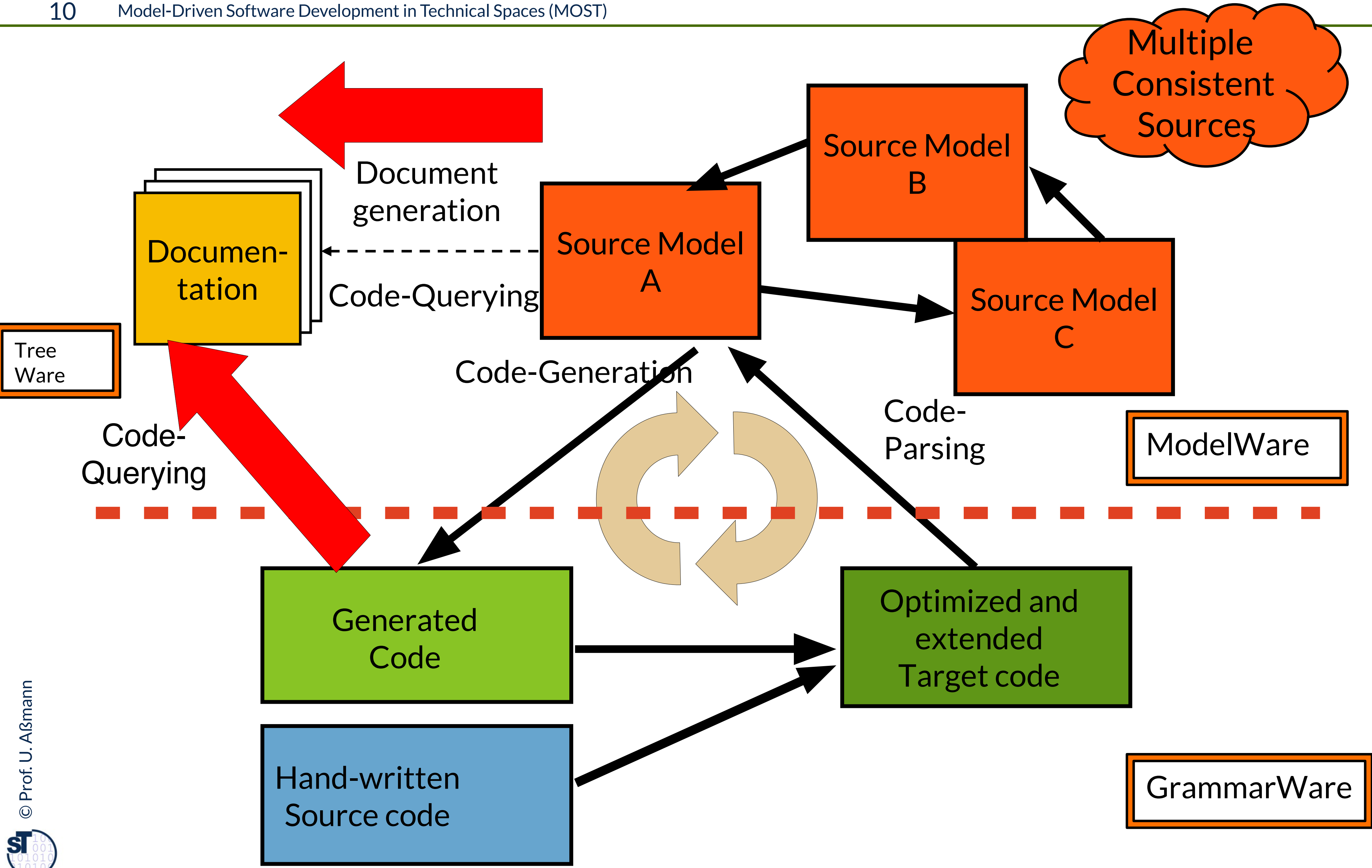
Tasks of Documentation Tools

- Basically, documentation generation is similar to code generation. Documentation is created in higher-order attributes on a link tree by a RAG
- **Documentation generation is an application areas for RAG**
- **Generation** of derived documents from code and models
  - Generation of Word (docx), LibreOffice (odt), rtf, xml, html formats
  - Generation of figures (svg, png, pdf)
  - Generation of snippets and generic snippets
  - Back-linking to originals
- **Filling** of documentation templates (with the hedge-principle)
- **Parameterization** with layouts
  - via css-style sheets

31.2 Generative, Template-Driven Documentation Tools

.. Documentation derived from code and models, based on template-based code generation

Macromodel Principle and Round-Trip Engineering



Documentation Tool JavaDoc is a Template Expander

- JavaDoc reads Java source code and extracts html from the code comments, based on **html templates**
  - Typical hedge-based code generation with generic snippets
- Generation of additional contents and indices
- Controlled by Java metadata attributes
  - @author, @date, @param
- Layouting via plugin classes called *doclets*
- JavaDoc has been realized for all programming languages

[JavaDoc; Wikipedia: http://de.wikipedia.org/wiki/JavaDoc]

JavaDoc is a Typical HRAG Application

- The html documentation is computed in a higher-order synthesized attribute `htmldoc:HTML`

```
// schematic, synthesis from bottom to top
Interpretation javaDoc(Tree → Tree)  {
  Attributions of Root(classes[]) {
    this.htmldoc := map + classes.htmldoc;
    <println(„Result is %S“, this.htmldoc)>
  }
  Attributions of Class(superclass:Class,methods{}) {
    this.htmldoc := <superclass.Name + methods.htmldoc;
  }
  Attributions of Method(name,comment) {
    this.htmldoc := „<h1>“+name+“</h2>“+comment.htmldoc;
  }
  Attributions of Comment(text) {
    this.htmldoc := text;
  }
}
```

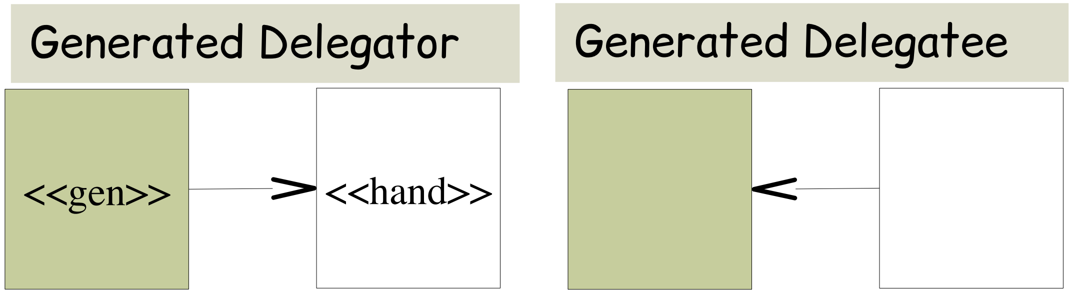
Composition of Separated Hand-Written and Generated Documentation Snippets

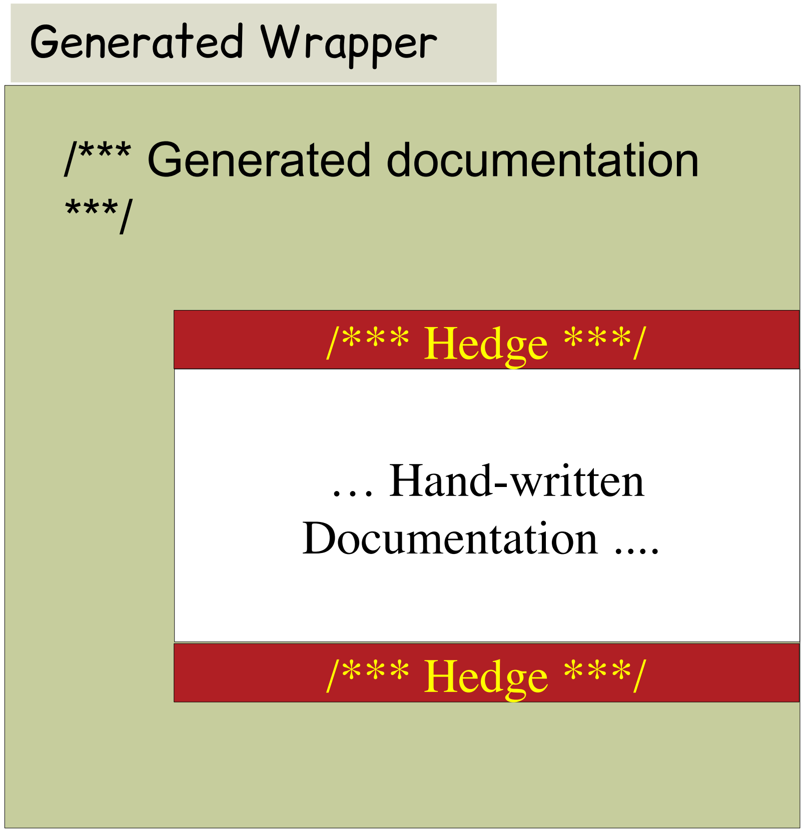
In separate files: Coupling by “include”

- Only possible if document format supports subdocument inclusion
  - e.g., TeX or Framemaker

In one file:

Coupling with **hegdes (Trennmarkierung)**

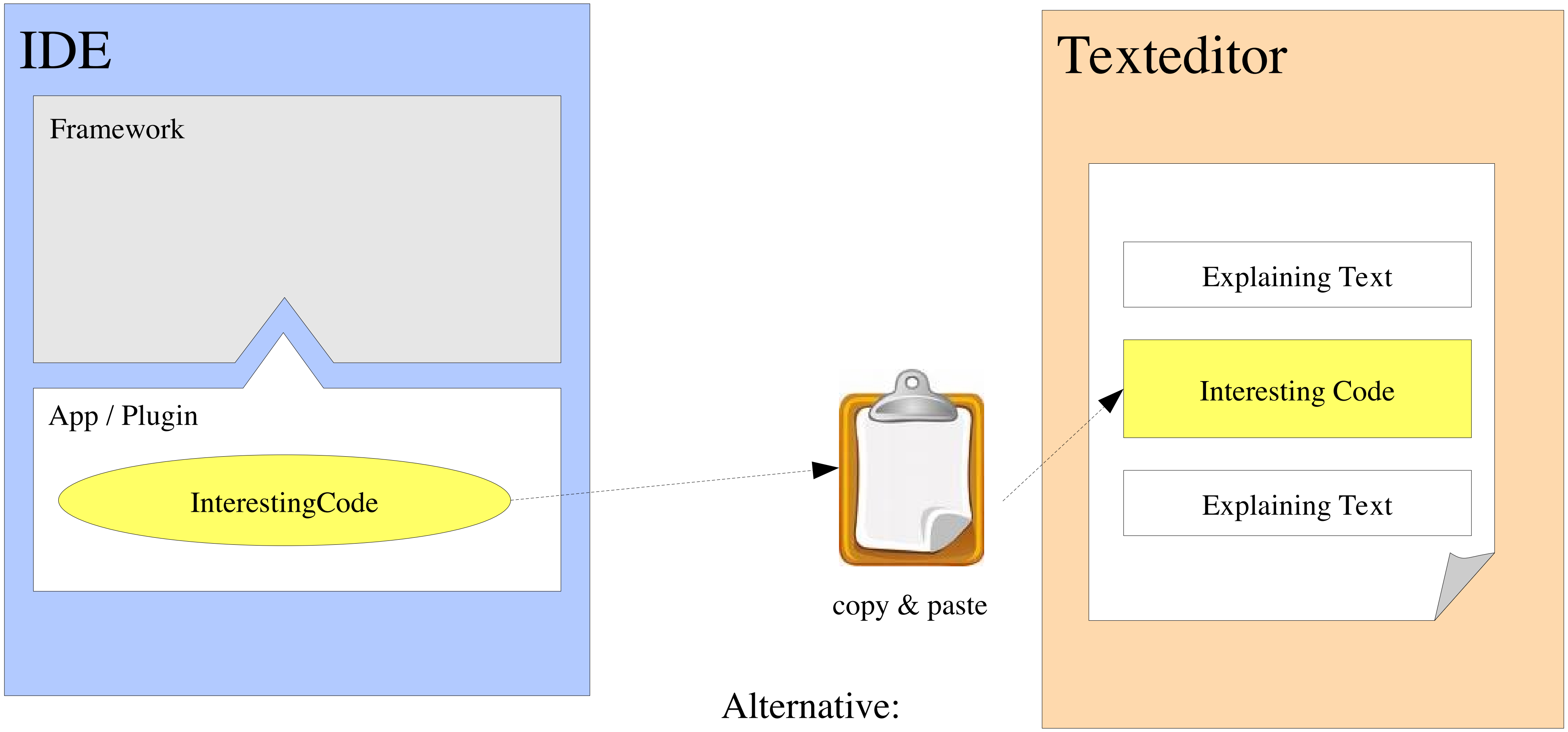



Generated Wrapper
/*** Generated documentation ***/
/*** Hedge ***/
… Hand-written Documentation ….
/*** Hedge ***/

31.3 Literate Programming

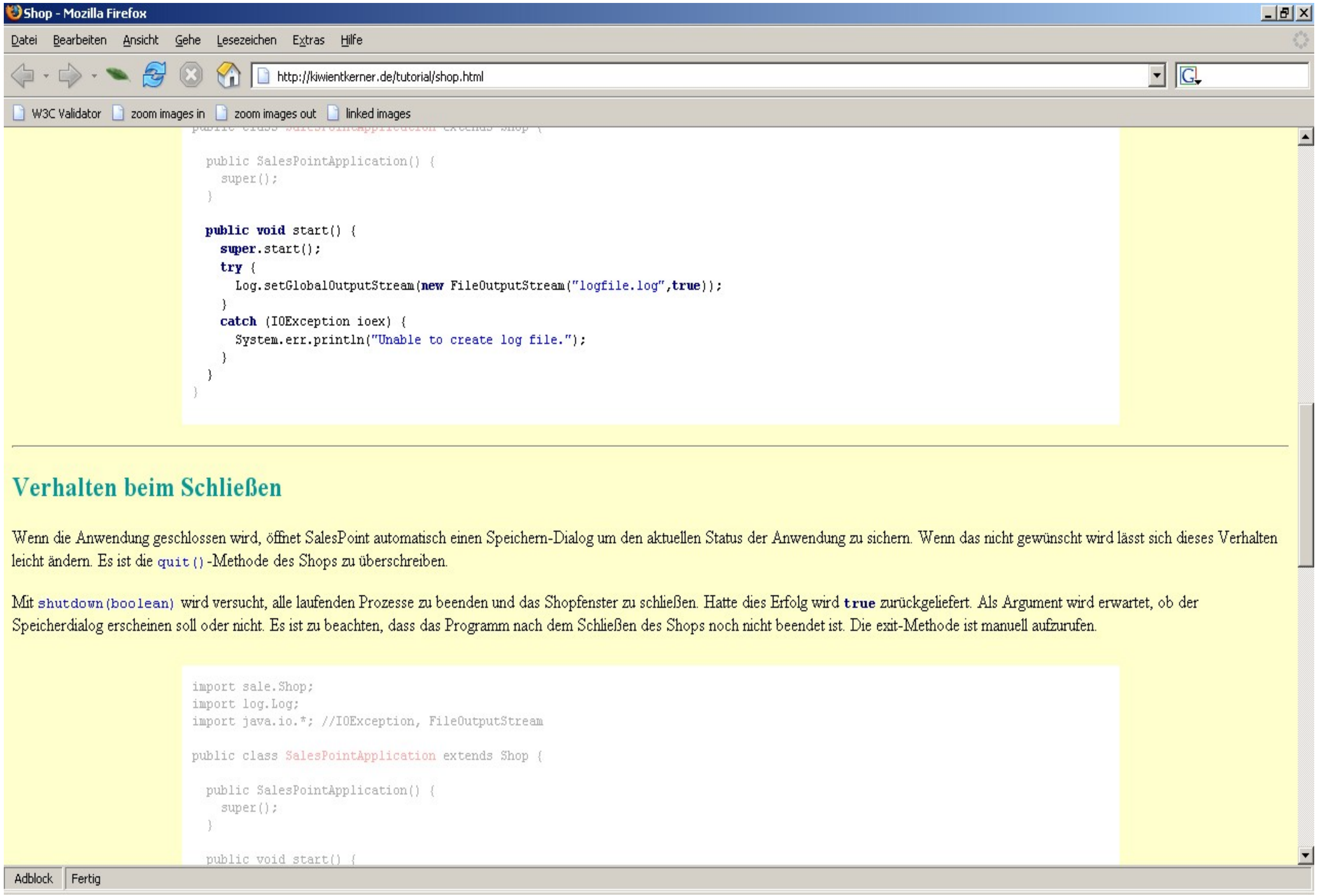
- They integrate code, models and documentation by **separating code from documentation**

Classic: Manual Writing of Tutorials



Alternative:
Code query
e.g., with
Xcerpt or QL

How to Write Integrated Documentation and Tutorials?


Shop - Mozilla Firefox
Datei Bearbeiten Ansicht Gehe Lesezeichen Extras Hilfe
http://kiwientkerner.de/tutorial/shop.html
W3C Validator zoom images in zoom images out linked images
Adblock Fertig

```
  public SalesPointApplication() {
    super();
  }

  public void start() {
    super.start();
    try {
      Log.setGlobalOutputStream(new FileOutputStream("logfile.log",true));
    }
    catch (IOException ioex) {
      System.err.println("Unable to create log file.");
    }
  }
}
```

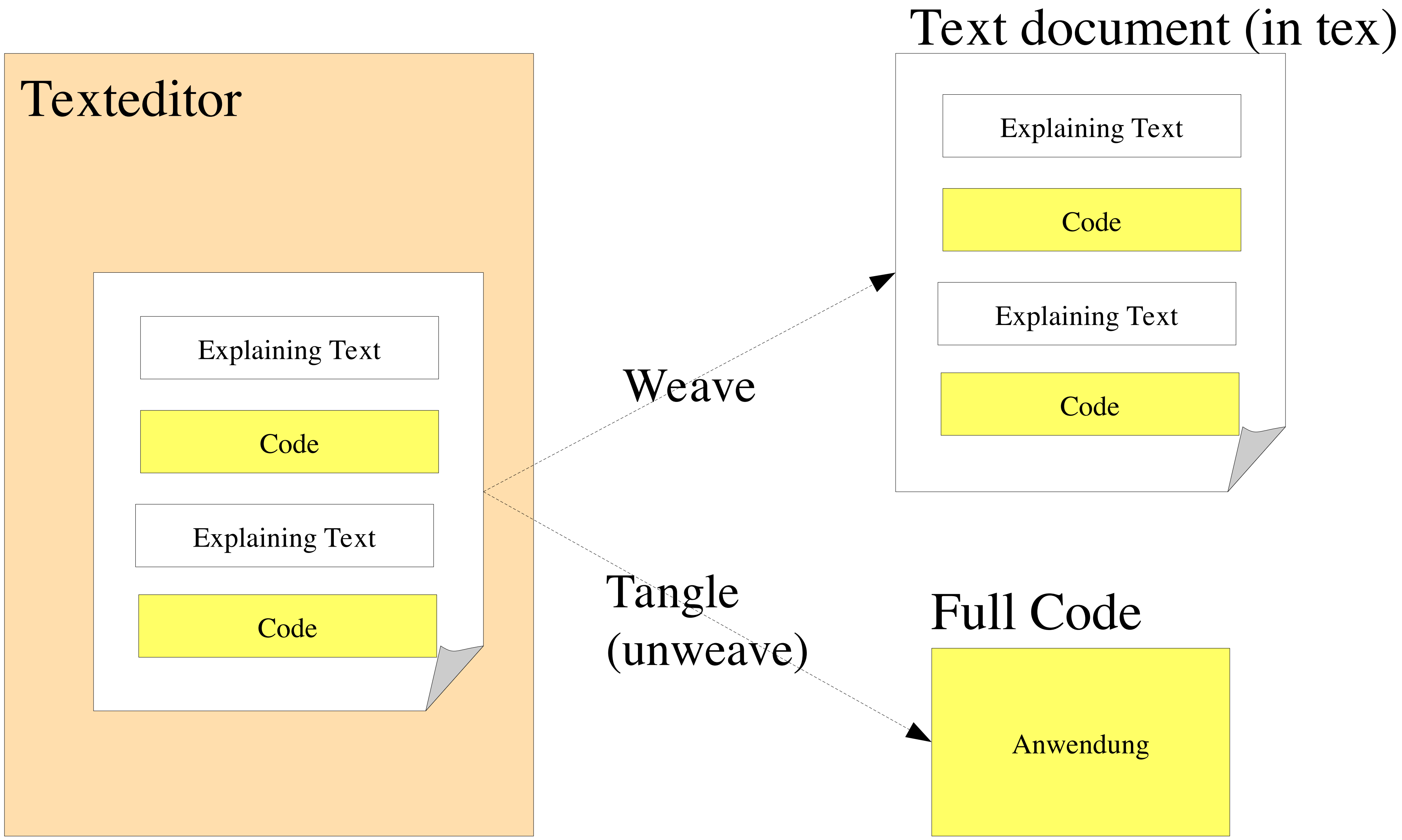
Verhalten beim Schließen

Wenn die Anwendung geschlossen wird, öffnet SalesPoint automatisch einen Speichern-Dialog um den aktuellen Status der Anwendung zu sichern. Wenn das nicht gewünscht wird lässt sich dieses Verhalten leicht ändern. Es ist die `quit()`-Methode des Shops zu überschreiben.

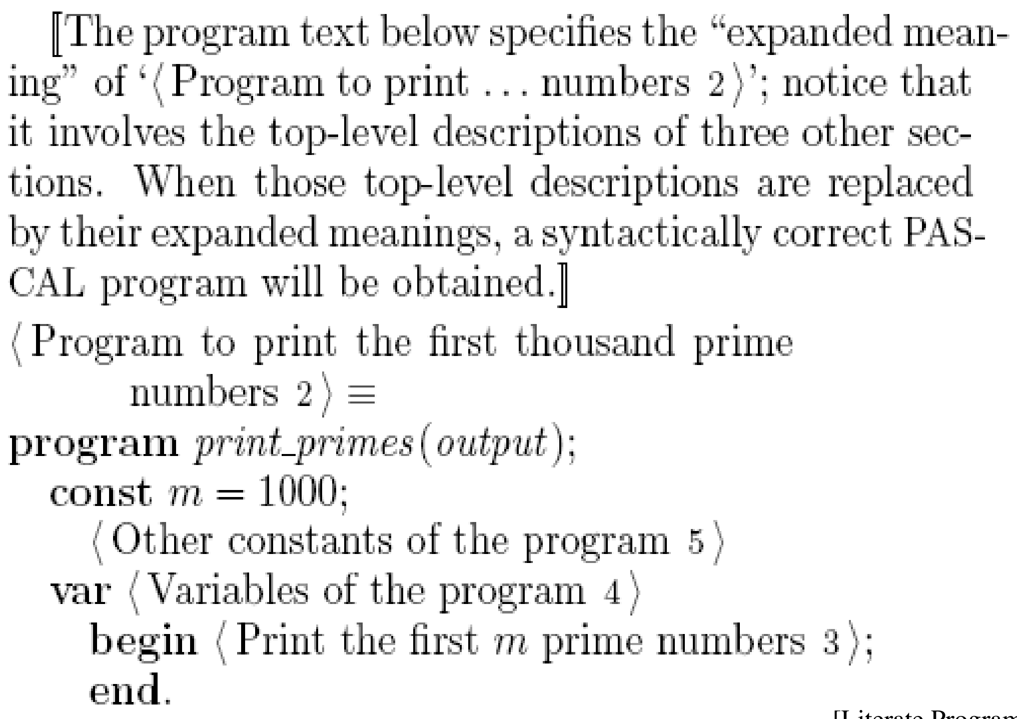
Mit `shutdown(boolean)` wird versucht, alle laufenden Prozesse zu beenden und das Shopfenster zu schließen. Hatte dies Erfolg wird **`true`** zurückgeliefert. Als Argument wird erwartet, ob der Speicherdialog erscheinen soll oder nicht. Es ist zu beachten, dass das Programm nach dem Schließen des Shops noch nicht beendet ist. Die exit-Methode ist manuell aufzurufen.

```
import sale.Shop;
import log.Log;
import java.io.*; //IOException, FileOutputStream

public class SalesPointApplication extends Shop {

  public SalesPointApplication() {
    super();
  }

  public void start() {
```

[Knuth] Literate Programming by Code Unweaving



Literate Programming



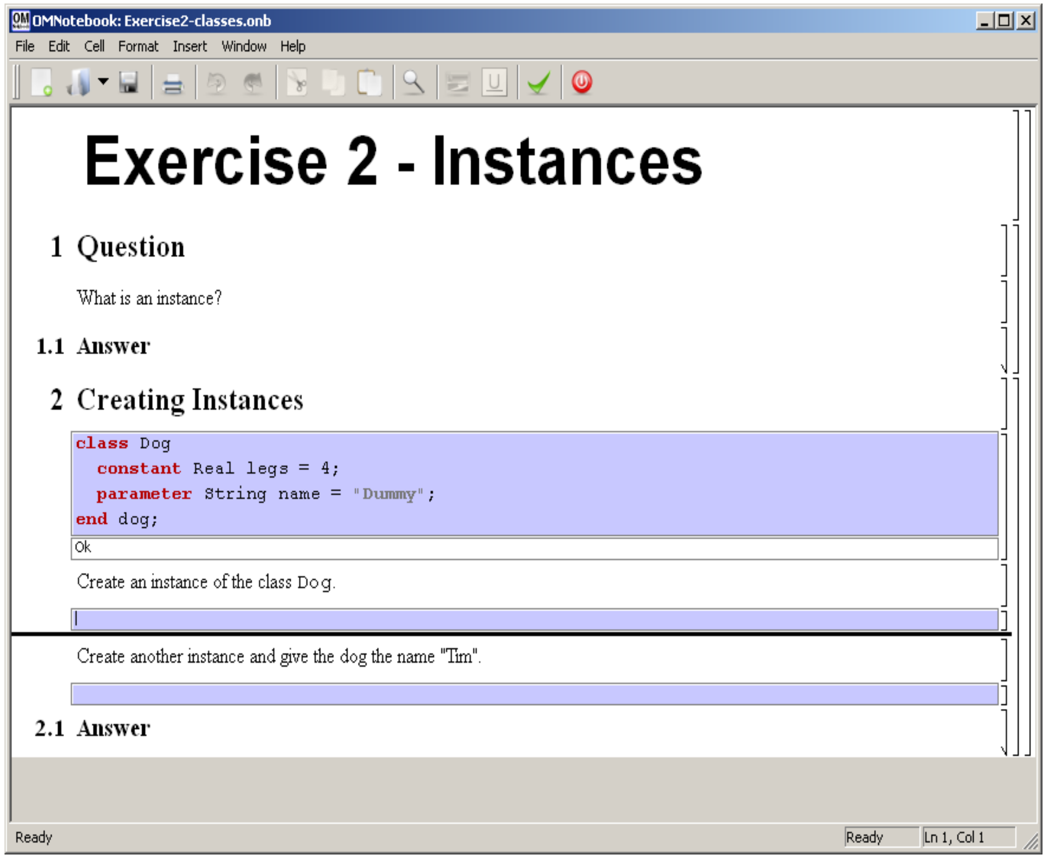
[The program text below specifies the “expanded meaning” of ‘⟨ Program to print . . . numbers 2 ⟩’; notice that it involves the top-level descriptions of three other sections. When those top-level descriptions are replaced by their expanded meanings, a syntactically correct PASCAL program will be obtained.]

```
⟨ Program to print the first thousand prime
        numbers 2 ⟩ ≡
program print_primes(output);
  const m = 1000;
    ⟨ Other constants of the program 5 ⟩
  var ⟨ Variables of the program 4 ⟩
    begin ⟨ Print the first m prime numbers 3 ⟩;
    end.
```

[Literate Programming
von Donald E. Knuth]

- The TeX engine is programmed literately
- Overview: http://www.literateprogramming.com/
- OMNotebook/DrModelica: http://www.modelica.org/tools

OMNotebook – Literate Programming with DrModelica


OMNotebook: Exercise2-classes.onb
File Edit Cell Format Insert Window Help
Ready
Ready
Ln 1, Col 1

Exercise 2 - Instances

1 Question

What is an instance?

1.1 Answer

2 Creating Instances

```
class Dog
  constant Real legs = 4;
  parameter String name = "Dummy";
end dog;
```

```
Ok
```

Create an instance of the class Dog.

Create another instance and give the dog the name "Tim".

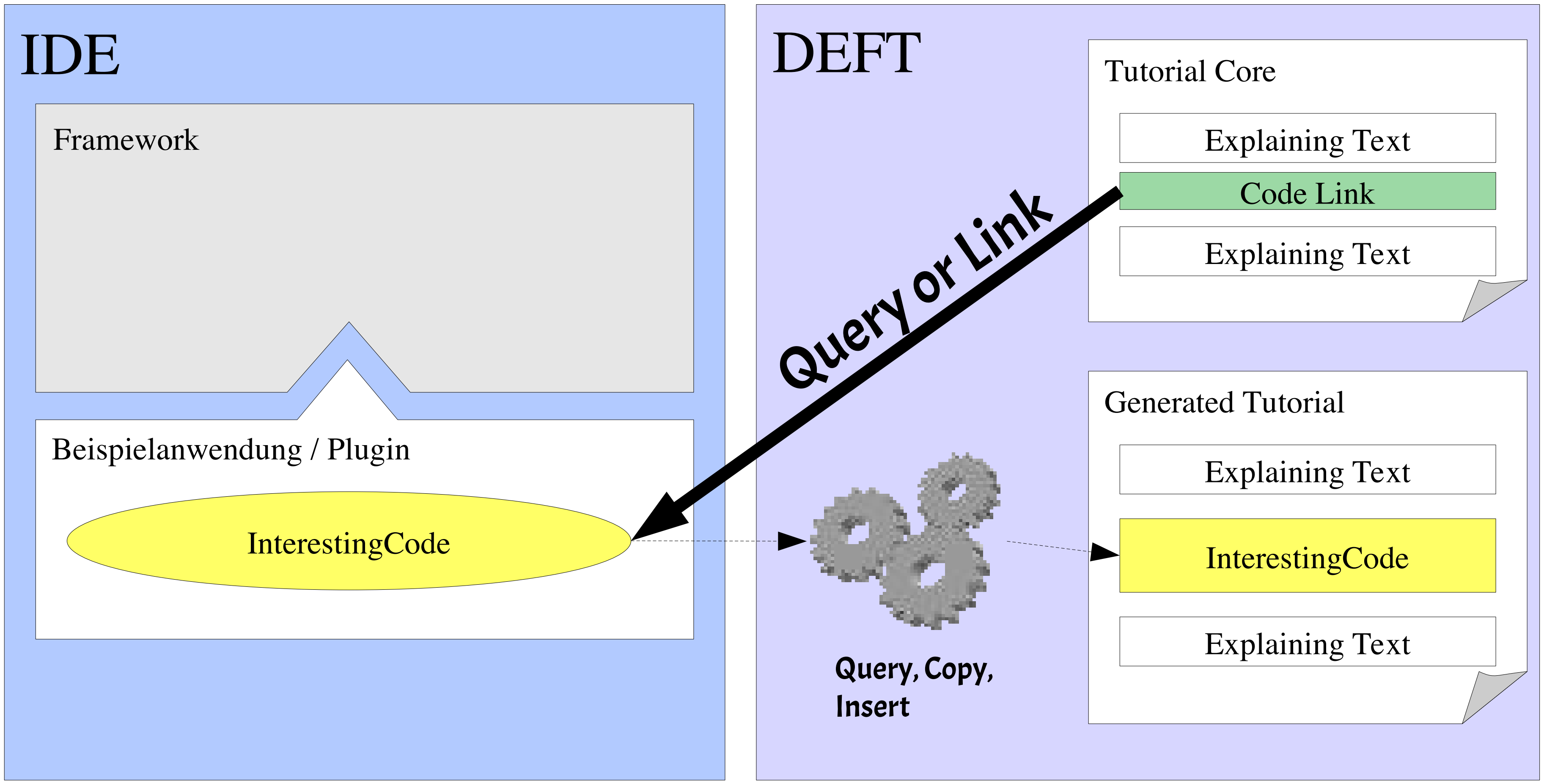
2.1 Answer

- Linked documents with interactive exercises
- Inspired by DrScheme und DrJava, learning tools for Scheme resp. Java
- www.openmodelica.org

31.4 Elucidative Documentation Tools

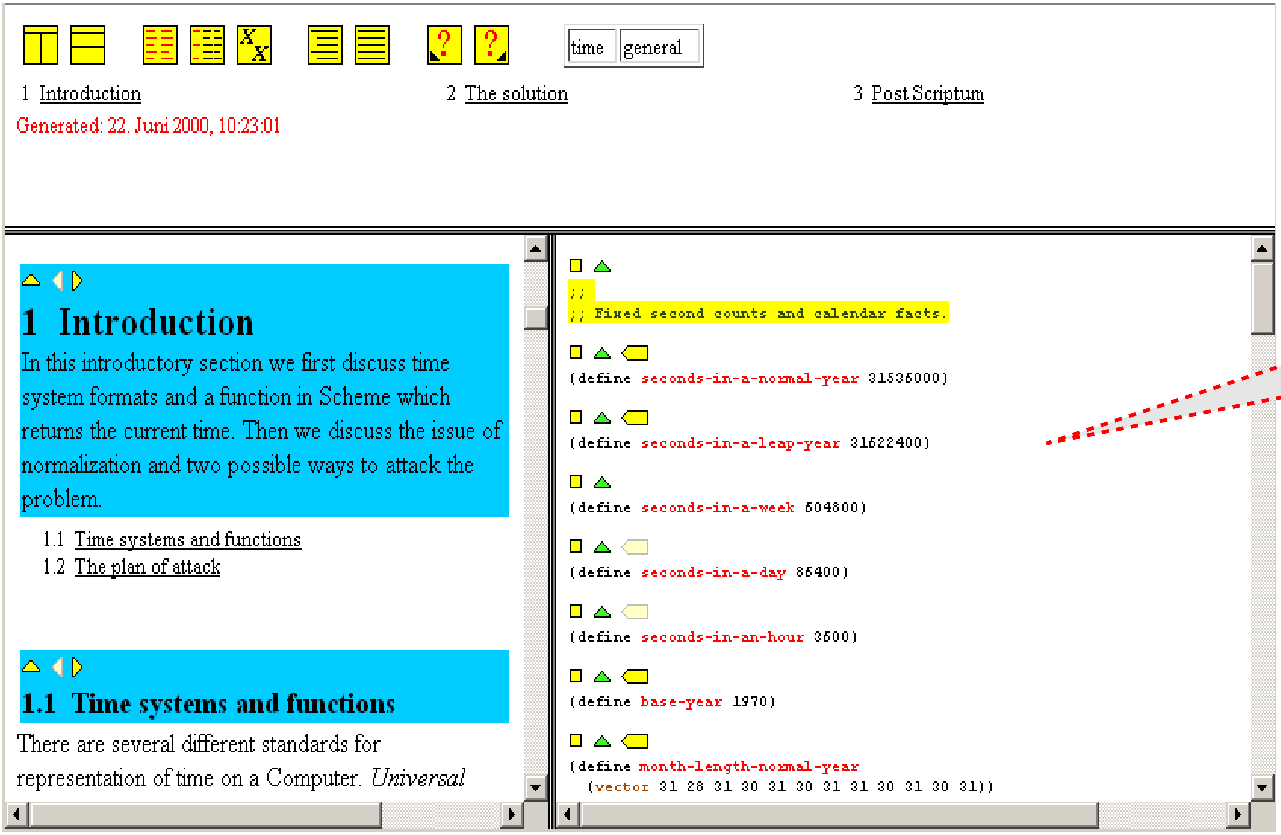
- They link code, models and documentation by **model and code mapping**
- **and renew the documentation by *hot updates***

Elucidative Programming Links Documentation with Queries to Code



hot update
(hot synchronisation):
round-trip engineering

Elucidative Programming



1 Introduction 2 The solution 3 Post Scriptum

Generated: 22. Juni 2000, 10:23:01

1 Introduction

In this introductory section we first discuss time system formats and a function in Scheme which returns the current time. Then we discuss the issue of normalization and two possible ways to attack the problem.

1.1 Time systems and functions
1.2 The plan of attack

1.1 Time systems and functions

There are several different standards for representation of time on a Computer. *Universal*

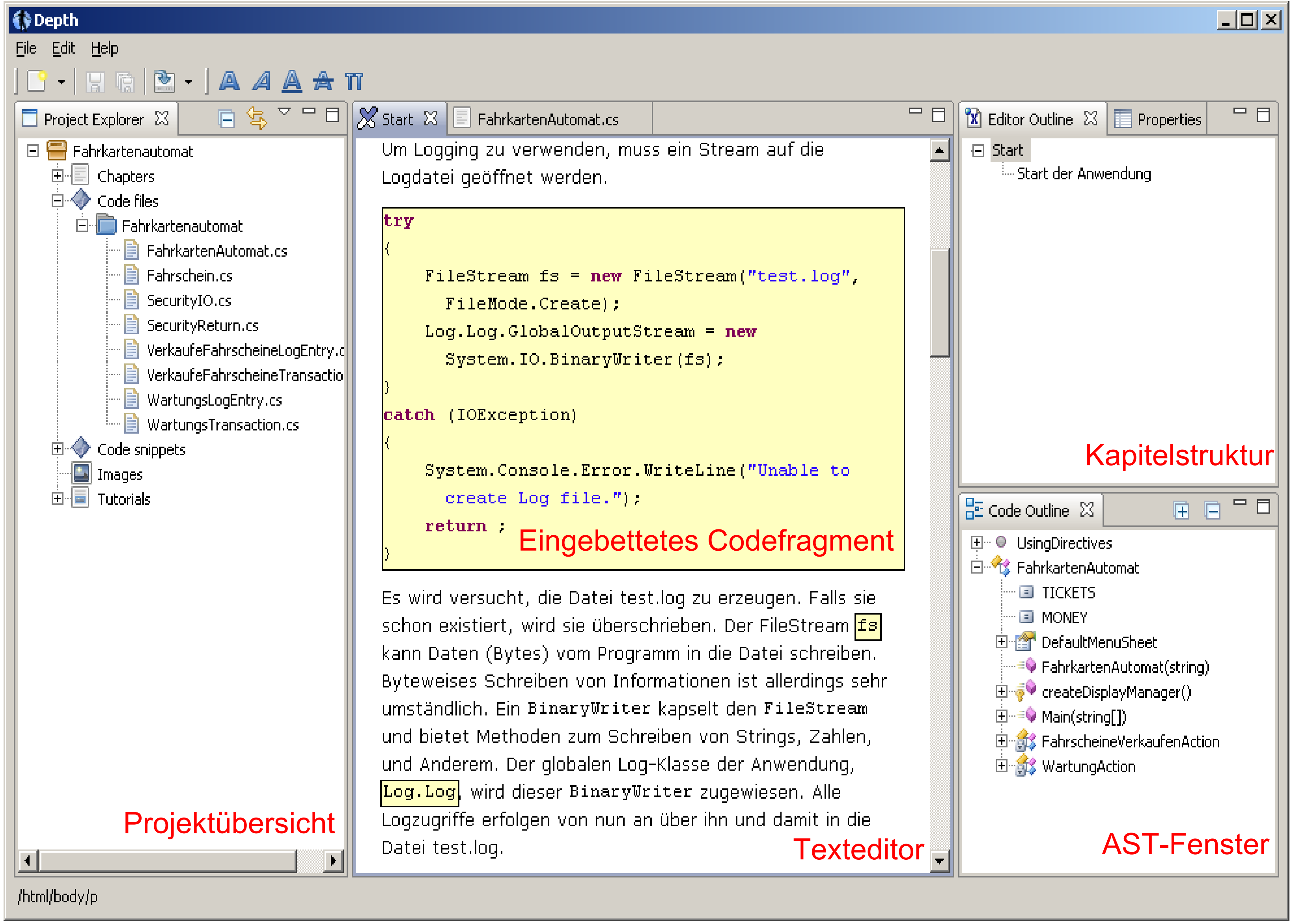
```
;;
;; Fixed second counts and calendar facts.

(define seconds-in-a-normal-year 31536000)

(define seconds-in-a-leap-year 31622400)

(define seconds-in-a-week 604800)

(define seconds-in-a-day 86400)

(define seconds-in-an-hour 3600)

(define base-year 1970)

(define month-length-normal-year
  (vector 31 28 31 30 31 30 31 31 30 31 30 31))
```

„Scheme Elucidator" Environment

- Elucidative Programming shows documentation and code in parallel
- http://www.cs.aau.dk/~normark/elucidative-programming/
- http://deftproject.org

hot update
(hot synchronisation):
round-trip engineering

Development Environment For Tutorials (DEFT www.deftproject.org)

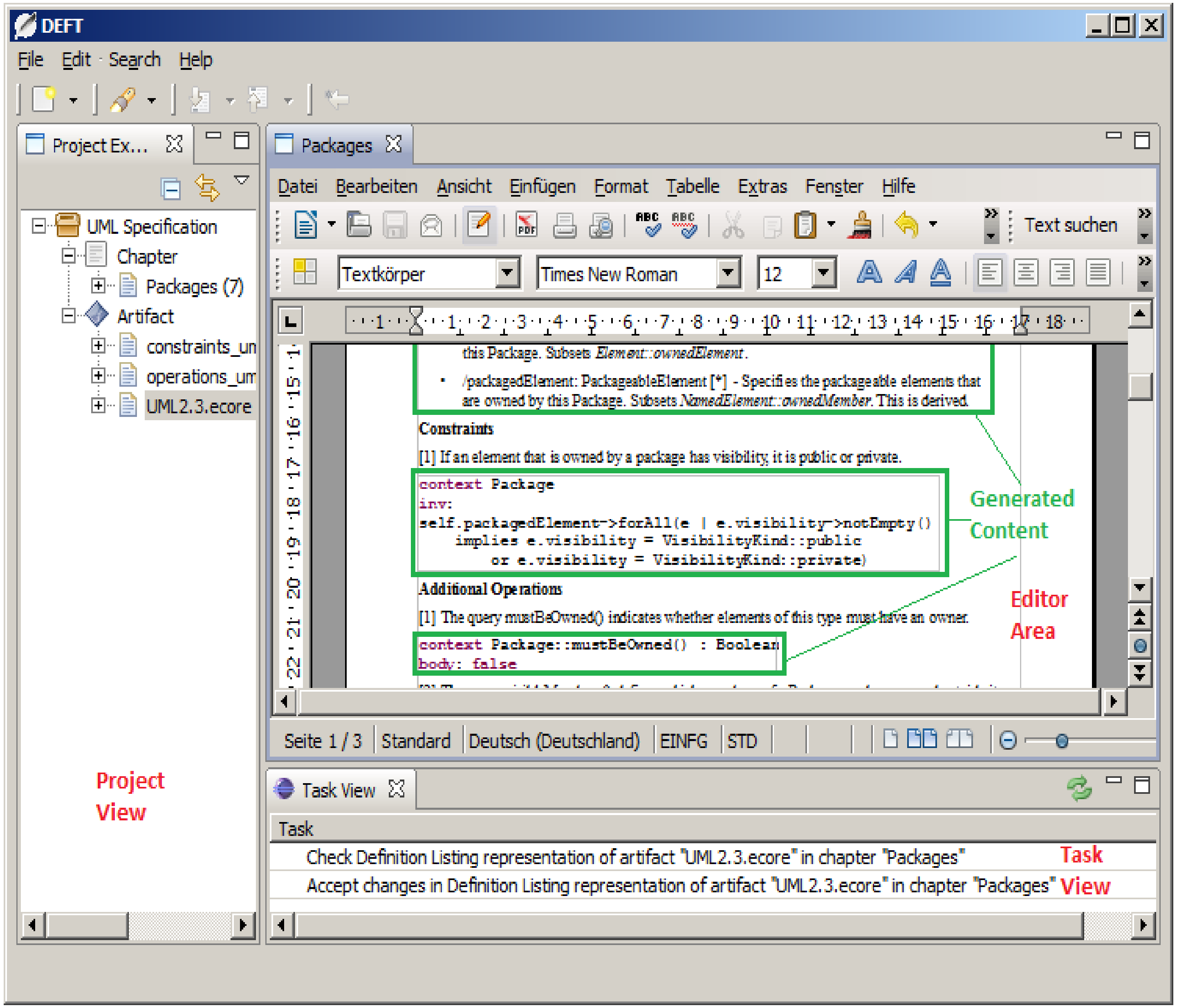


Um Logging zu verwenden, muss ein Stream auf die Logdatei geöffnet werden.

```
try
{
    FileStream fs = new FileStream("test.log",
      FileMode.Create);
    Log.Log.GlobalOutputStream = new
      System.IO.BinaryWriter(fs);
}
catch (IOException)
{
    System.Console.Error.WriteLine("Unable to
      create Log file.");
    return ;
}
```

Es wird versucht, die Datei test.log zu erzeugen. Falls sie schon existiert, wird sie überschrieben. Der FileStream `fs` kann Daten (Bytes) vom Programm in die Datei schreiben. Byteweises Schreiben von Informationen ist allerdings sehr umständlich. Ein `BinaryWriter` kapselt den `FileStream` und bietet Methoden zum Schreiben von Strings, Zahlen, und Anderem. Der globalen Log-Klasse der Anwendung, `Log.Log`, wird dieser `BinaryWriter` zugewiesen. Alle Logzugriffe erfolgen von nun an über ihn und damit in die Datei test.log.

Embedding UML Constraints for UML Models into Documentation



this Package. Subsets *Element::ownedElement*.

- /packagedElement: PackageableElement [*] - Specifies the packageable elements that are owned by this Package. Subsets *NamedElement::ownedMember*. This is derived.

Constraints

[1] If an element that is owned by a package has visibility, it is public or private.

```
context Package
inv:
self.packagedElement->forAll(e | e.visibility->notEmpty()
    implies e.visibility = VisibilityKind::public
        or e.visibility = VisibilityKind::private)
```

Additional Operations

[1] The query mustBeOwned() indicates whether elements of this type must have an owner.

```
context Package::mustBeOwned() : Boolean
body: false
```

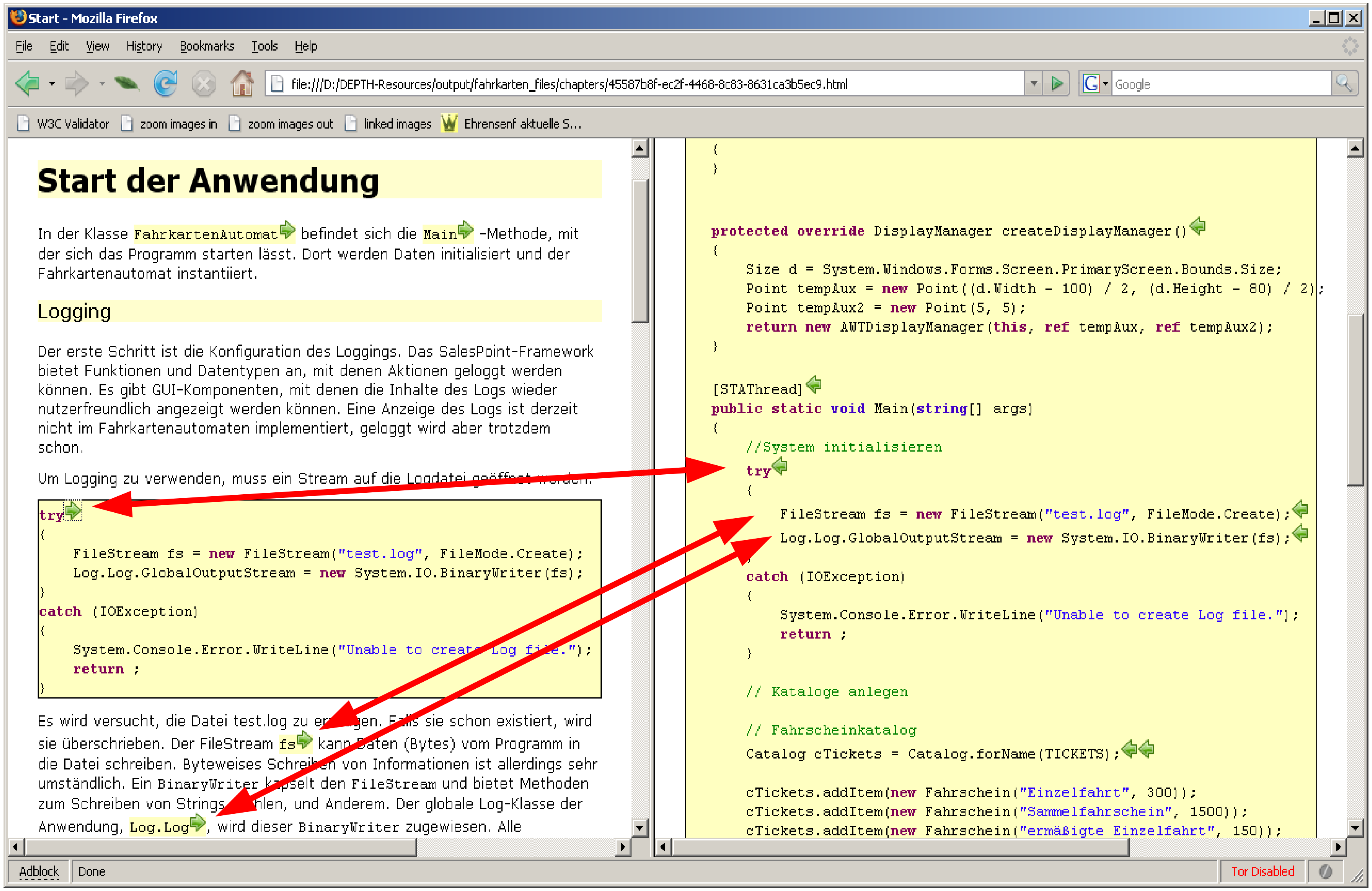
Check Definition Listing representation of artifact "UML2.3.ecore" in chapter "Packages"

Accept changes in Definition Listing representation of artifact "UML2.3.ecore" in chapter "Packages"

Development Environment For Tutorials (DEFT)

- Eclipse RCP application, language independent
- Management of code, models and text
- Prettyprinting of code fragments from code templates
- Hot update of generated documentation
  - Automatic update of embedded code fragments
  - Notiviation if code fragments have changed

Generated HTML Tutorial


Start - Mozilla Firefox
File Edit View History Bookmarks Tools Help
file:///D:/DEPTH-Resources/output/fahrkarten_files/chapters/45587b8f-ec2f-4468-8c83-8631ca3b5ec9.html
Google
W3C Validator
zoom images in
zoom images out
linked images
Ehrensenf aktuelle S...
Adblock
Done
Tor Disabled

Start der Anwendung

In der Klasse `FahrkartenAutomat` befindet sich die `Main` -Methode, mit der sich das Programm starten lässt. Dort werden Daten initialisiert und der Fahrkartenautomat instantiiert.

Logging

Der erste Schritt ist die Konfiguration des Loggings. Das SalesPoint-Framework bietet Funktionen und Datentypen an, mit denen Aktionen geloggt werden können. Es gibt GUI-Komponenten, mit denen die Inhalte des Logs wieder nutzerfreundlich angezeigt werden können. Eine Anzeige des Logs ist derzeit nicht im Fahrkartenautomaten implementiert, geloggt wird aber trotzdem schon.

Um Logging zu verwenden, muss ein Stream auf die Logdatei geöffnet werden.

```
try
{
    FileStream fs = new FileStream("test.log", FileMode.Create);
    Log.Log.GlobalOutputStream = new System.IO.BinaryWriter(fs);
}
catch (IOException)
{
    System.Console.Error.WriteLine("Unable to create Log file.");
    return ;
}
```

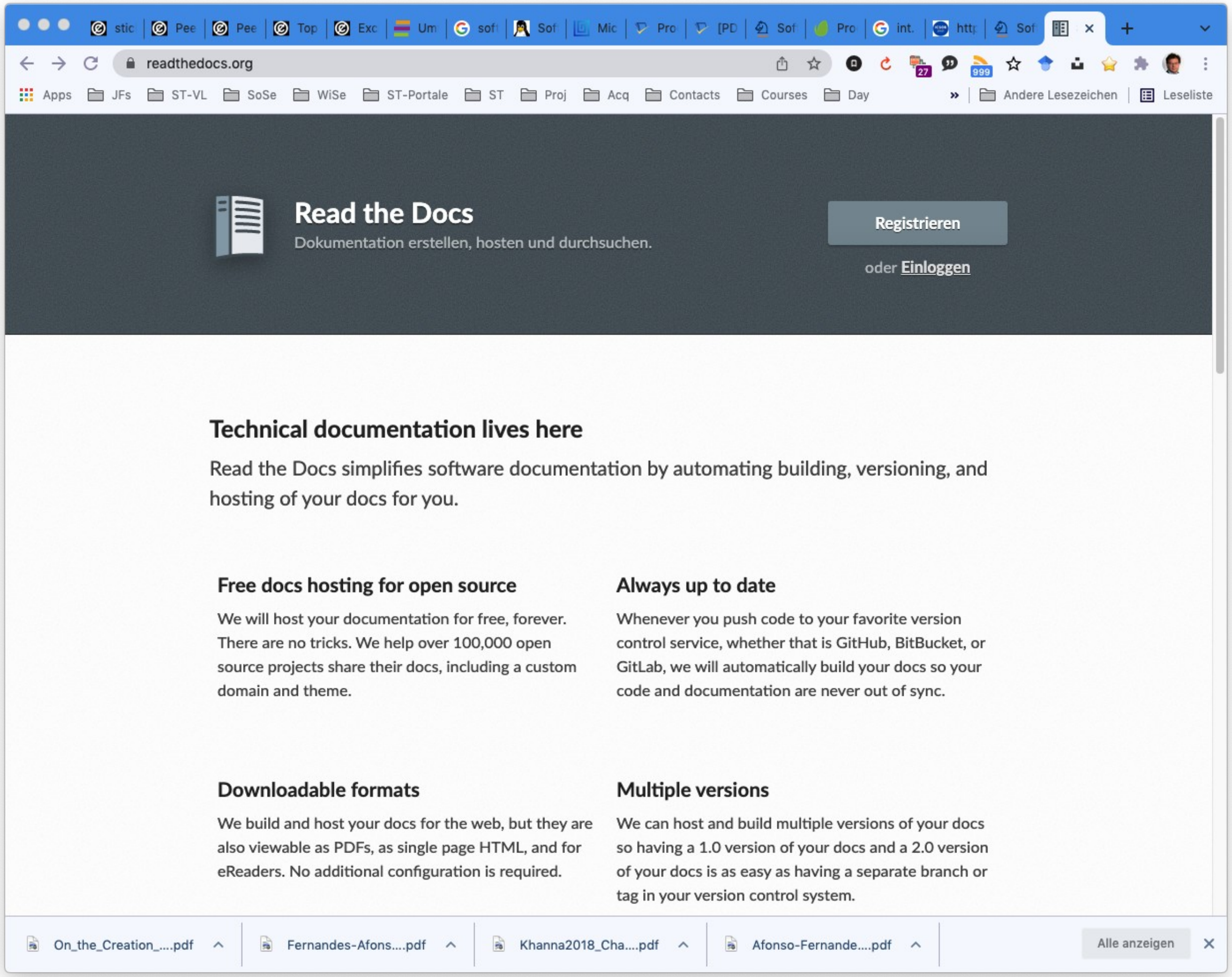
Es wird versucht, die Datei test.log zu erzeugen. Falls sie schon existiert, wird sie überschrieben. Der FileStream `fs` kann Daten (Bytes) vom Programm in die Datei schreiben. Byteweises Schreiben von Informationen ist allerdings sehr umständlich. Ein `BinaryWriter` kapselt den `FileStream` und bietet Methoden zum Schreiben von Strings, Zahlen, und Anderem. Der globale Log-Klasse der Anwendung, `Log.Log`, wird dieser `BinaryWriter` zugewiesen. Alle

```
{
}


protected override DisplayManager createDisplayManager()
{
    Size d = System.Windows.Forms.Screen.PrimaryScreen.Bounds.Size;
    Point tempAux = new Point((d.Width - 100) / 2, (d.Height - 80) / 2);
    Point tempAux2 = new Point(5, 5);
    return new AWTDisplayManager(this, ref tempAux, ref tempAux2);
}

[STAThread]
public static void Main(string[] args)
{
    //System initialisieren
    try
    {
        FileStream fs = new FileStream("test.log", FileMode.Create);
        Log.Log.GlobalOutputStream = new System.IO.BinaryWriter(fs);
    }
    catch (IOException)
    {
        System.Console.Error.WriteLine("Unable to create Log file.");
        return ;
    }

    // Kataloge anlegen

    // Fahrscheinkatalog
    Catalog cTickets = Catalog.forName(TICKETS);

    cTickets.addItem(new Fahrschein("Einzelfahrt", 300));
    cTickets.addItem(new Fahrschein("Sammelfahrschein", 1500));
    cTickets.addItem(new Fahrschein("ermäßigte Einzelfahrt", 150));
```

31.5 Web-based Documentation Generators based on Markdown

Sphinx and the Documentation Cloud readthedocs.org

- readthedocs is a cloud for documentation projects
- supporting two documentation generators *sphinx* and *mkdocs*



Read the Docs

Dokumentation erstellen, hosten und durchsuchen.

Technical documentation lives here

Read the Docs simplifies software documentation by automating building, versioning, and hosting of your docs for you.

Free docs hosting for open source

We will host your documentation for free, forever. There are no tricks. We help over 100,000 open source projects share their docs, including a custom domain and theme.

Always up to date

Whenever you push code to your favorite version control service, whether that is GitHub, BitBucket, or GitLab, we will automatically build your docs so your code and documentation are never out of sync.

Downloadable formats

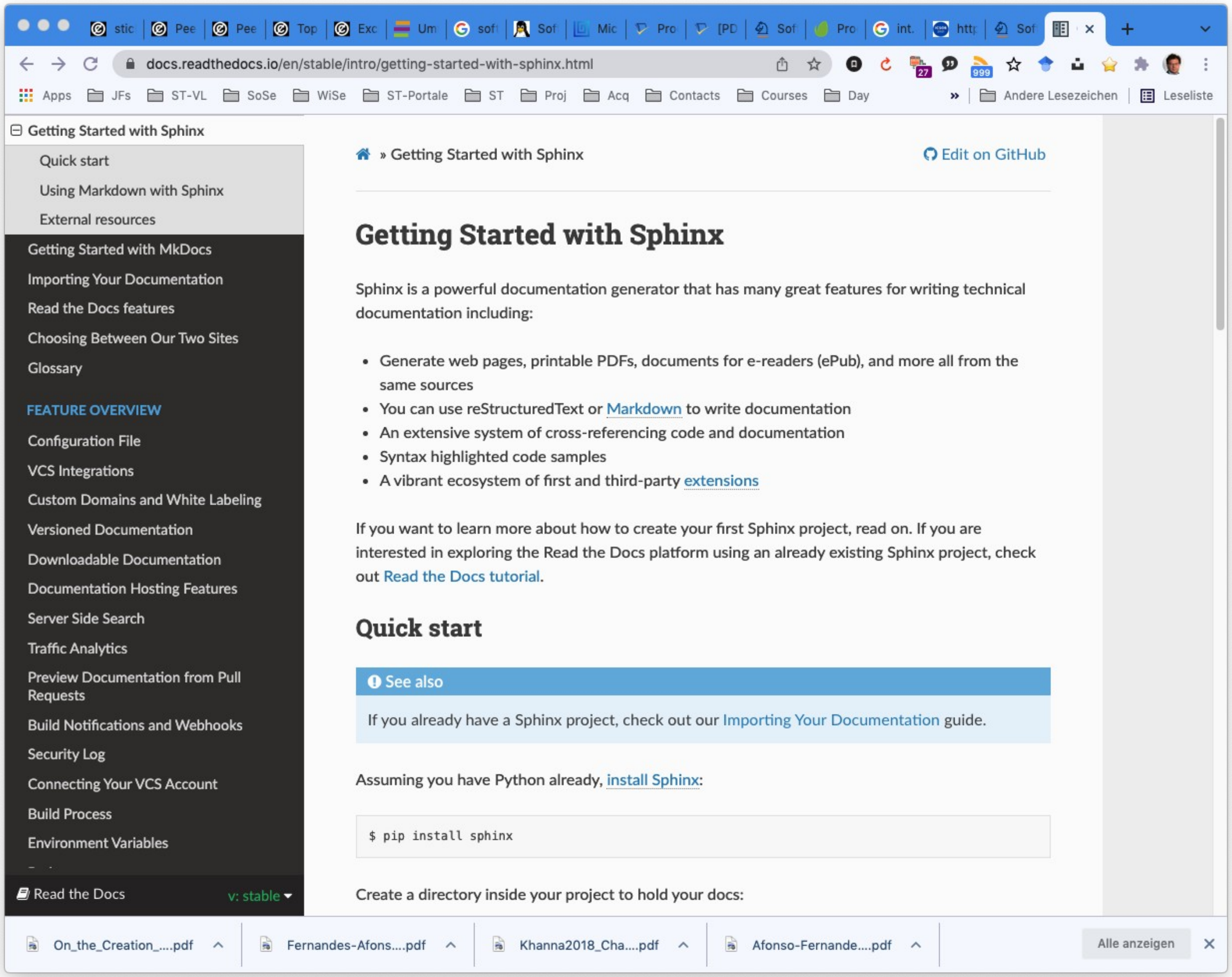
We build and host your docs for the web, but they are also viewable as PDFs, as single page HTML, and for eReaders. No additional configuration is required.

Multiple versions

We can host and build multiple versions of your docs so having a 1.0 version of your docs and a 2.0 version of your docs is as easy as having a separate branch or tag in your version control system.

Sphinx

- Architecture documentation
- User documentation
- Files in formats restructuredText and Markdown are transformed to HTML
- Treats entire directories
- many output formats (e.g., Latex)
- Can be coupled with Javadoc or similar API doc generators



Getting Started with Sphinx

Sphinx is a powerful documentation generator that has many great features for writing technical documentation including:

- Generate web pages, printable PDFs, documents for e-readers (ePub), and more all from the same sources
- You can use reStructuredText or Markdown to write documentation
- An extensive system of cross-referencing code and documentation
- Syntax highlighted code samples
- A vibrant ecosystem of first and third-party extensions

If you want to learn more about how to create your first Sphinx project, read on. If you are interested in exploring the Read the Docs platform using an already existing Sphinx project, check out Read the Docs tutorial.

Quick start

See also

If you already have a Sphinx project, check out our Importing Your Documentation guide.

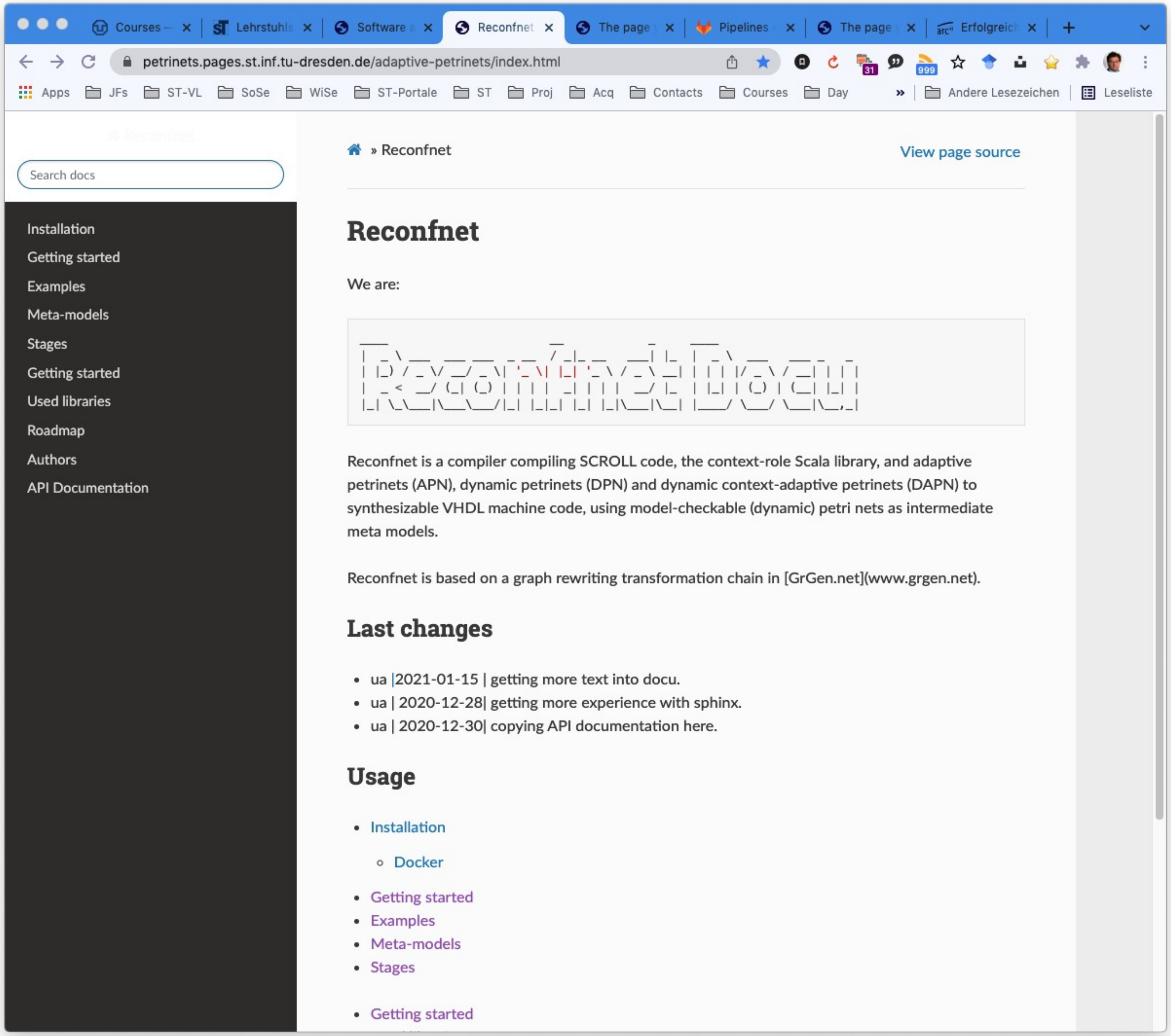
Assuming you have Python already, install Sphinx:

```
$ pip install sphinx
```

Create a directory inside your project to hold your docs:

Example Sphinx Project

- Petrinet compiler Reconfnet https://petrinets.pages.st.inf.tu-dresden.de/adaptive-petrinets/index.html
-



Reconfnet

We are:

```
 ____                        __            _     ____
|  _ \ ___  ___ ___  _ __  / _|_ __   ___| |_  |  _ \  ___   ___ _   _
| |_) / _ \/ __/ _ \| '_ \| |_| '_ \ / _ \ __| | | | |/ _ \ / __| | | |
|  _ <  __/ (_| (_) | | | |  _| | | |  __/ |_  | |_| | (_) | (__| |_| |
|_| \_\___|\___\___/|_| |_|_| |_| |_|\___|\__| |____/ \___/ \___|\__,_|
```

Reconfnet is a compiler compiling SCROLL code, the context-role Scala library, and adaptive petrinets (APN), dynamic petrinets (DPN) and dynamic context-adaptive petrinets (DAPN) to synthesizable VHDL machine code, using model-checkable (dynamic) petri nets as intermediate meta models.

Reconfnet is based on a graph rewriting transformation chain in [GrGen.net](www.grgen.net).

Last changes

- ua |2021-01-15 | getting more text into docu.
- ua | 2020-12-28| getting more experience with sphinx.
- ua | 2020-12-30| copying API documentation here.

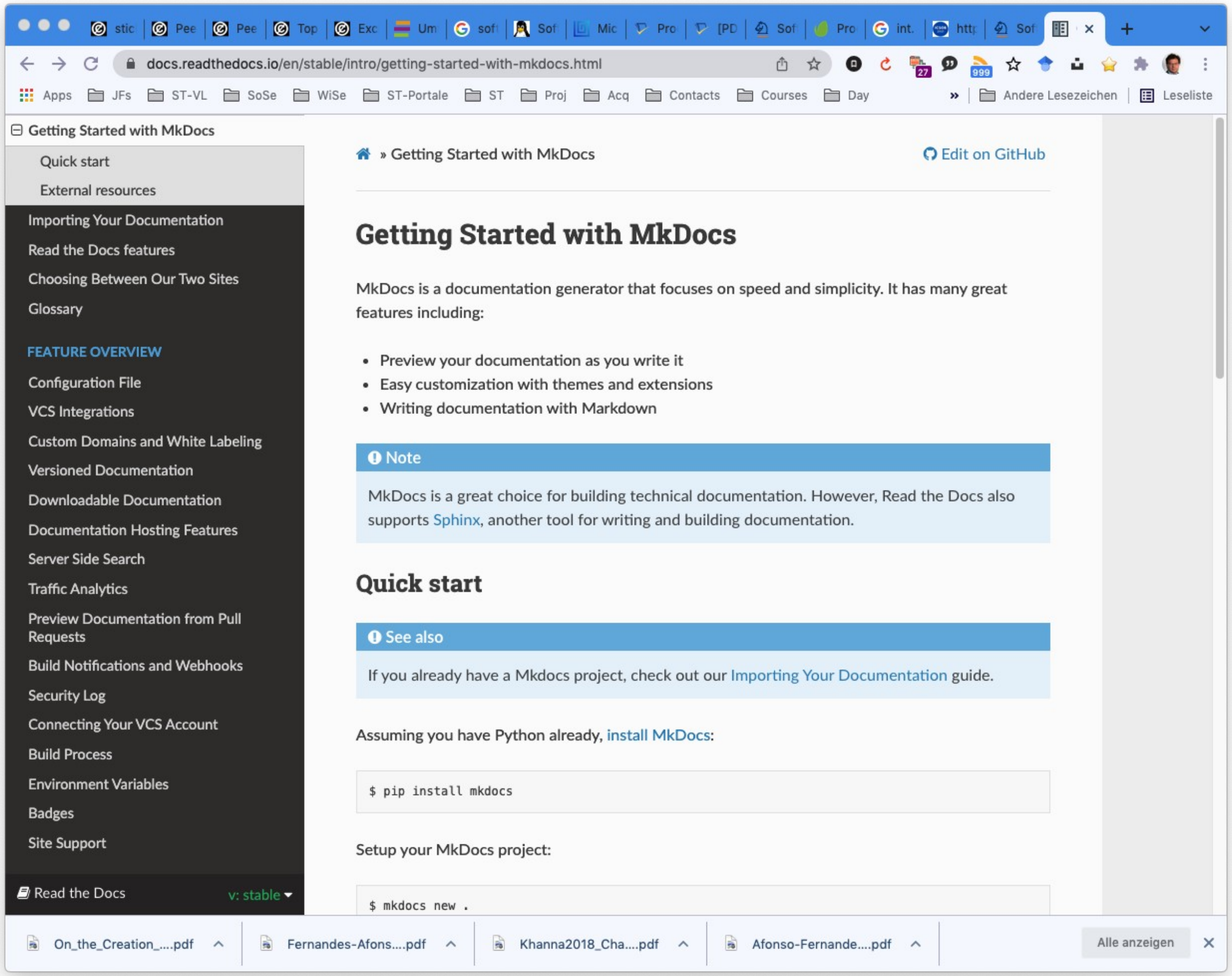
Usage

- Installation
  - Docker
- Getting started
- Examples
- Meta-models
- Stages

- Getting started

MkDocs

- Markdown files to HTML files
- several output formats


docs.readthedocs.io/en/stable/intro/getting-started-with-mkdocs.html
Apps JFs ST-VL SoSe WiSe ST-Portale ST Proj Acq Contacts Courses Day Andere Lesezeichen Leseliste
Getting Started with MkDocs
Quick start
External resources
Importing Your Documentation
Read the Docs features
Choosing Between Our Two Sites
Glossary
FEATURE OVERVIEW
Configuration File
VCS Integrations
Custom Domains and White Labeling
Versioned Documentation
Downloadable Documentation
Documentation Hosting Features
Server Side Search
Traffic Analytics
Preview Documentation from Pull Requests
Build Notifications and Webhooks
Security Log
Connecting Your VCS Account
Build Process
Environment Variables
Badges
Site Support
Read the Docs v: stable
» Getting Started with MkDocs
Edit on GitHub
On_the_Creation_....pdf
Fernandes-Afons....pdf
Khanna2018_Cha....pdf
Afonso-Fernande....pdf
Alle anzeigen

Getting Started with MkDocs

MkDocs is a documentation generator that focuses on speed and simplicity. It has many great features including:

- Preview your documentation as you write it
- Easy customization with themes and extensions
- Writing documentation with Markdown

Note

MkDocs is a great choice for building technical documentation. However, Read the Docs also supports Sphinx, another tool for writing and building documentation.

Quick start

See also

If you already have a Mkdocs project, check out our Importing Your Documentation guide.

Assuming you have Python already, install MkDocs:

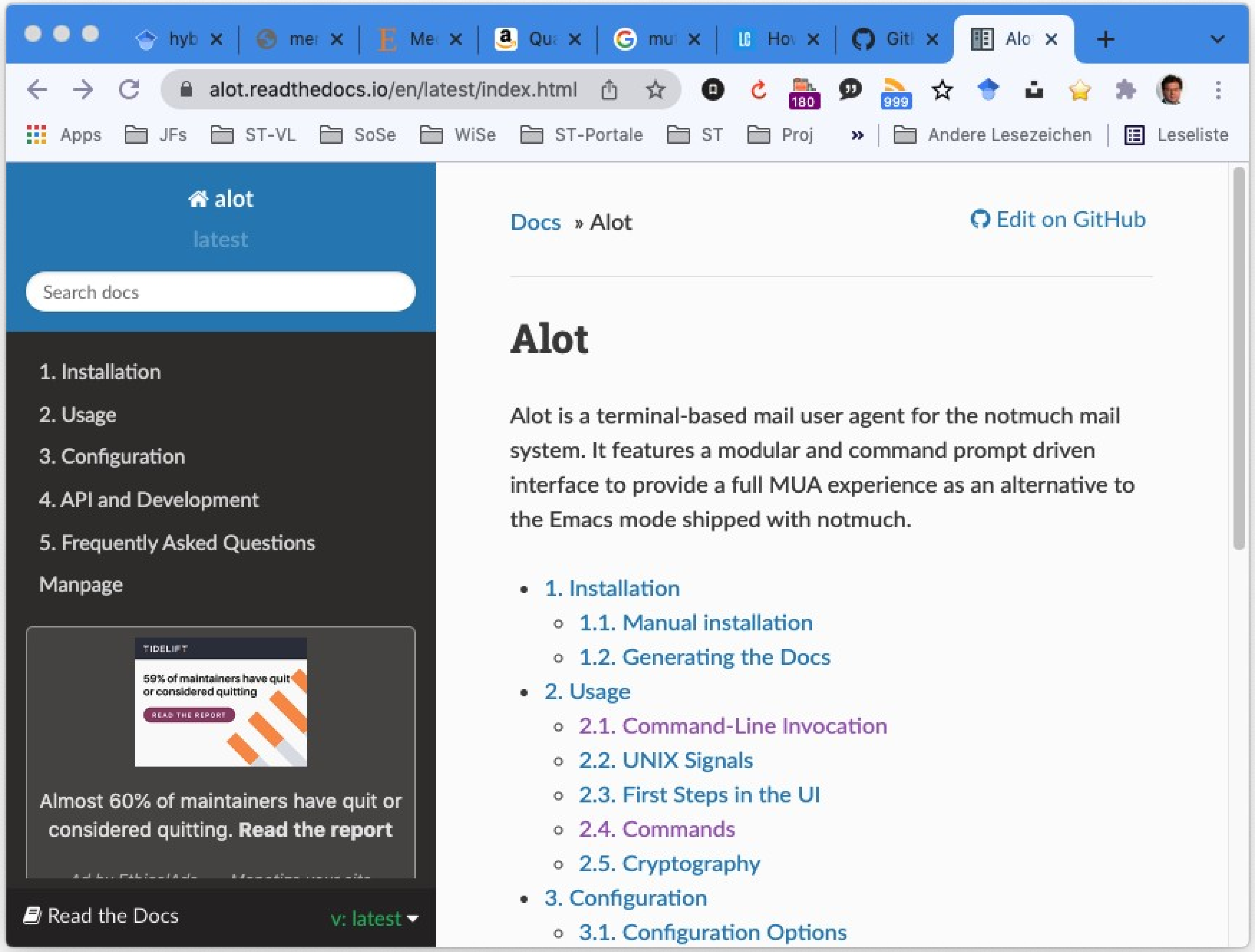
```
$ pip install mkdocs
```

Setup your MkDocs project:

```
$ mkdocs new .
```

Alot – a Mail User Agent Documented on readthedocs

- https://alot.readthedocs.io/



Docs » Alot

Edit on GitHub

Alot

Alot is a terminal-based mail user agent for the notmuch mail system. It features a modular and command prompt driven interface to provide a full MUA experience as an alternative to the Emacs mode shipped with notmuch.

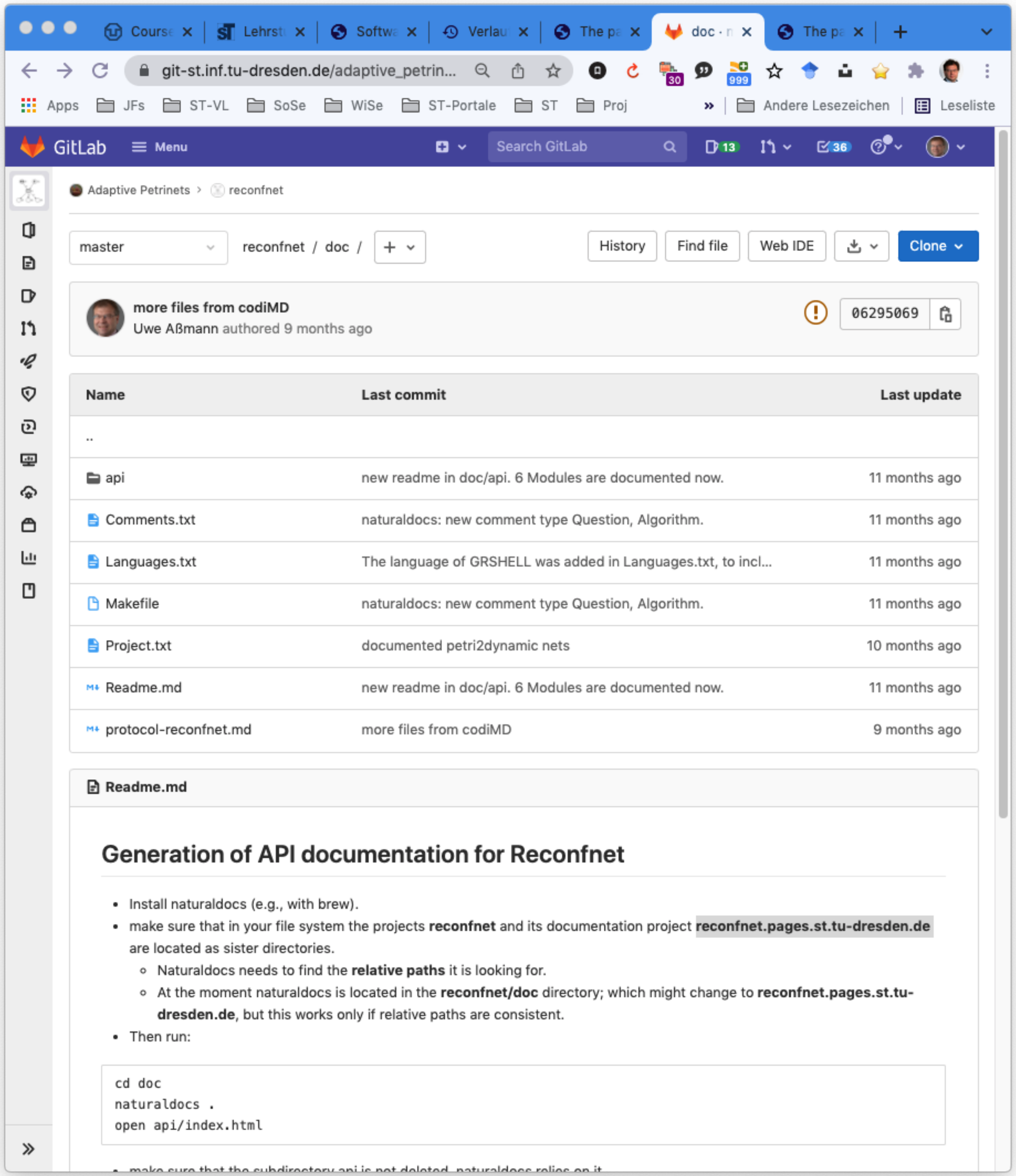
- 1. Installation
  - 1.1. Manual installation
  - 1.2. Generating the Docs
- 2. Usage
  - 2.1. Command-Line Invocation
  - 2.2. UNIX Signals
  - 2.3. First Steps in the UI
  - 2.4. Commands
  - 2.5. Cryptography
- 3. Configuration
  - 3.1. Configuration Options

1. Installation
2. Usage
3. Configuration
4. API and Development
5. Frequently Asked Questions
Manpage

Almost 60% of maintainers have quit or considered quitting. **Read the report**

NaturalDocs Generic API Documentation Generator

- Similar to JavaDoc, but more than 20 languages
- own keywords can be defined
- Example gitlab project from which API documentation for GrGen can be generated
  - https://git-st.inf.tu-dresden.de/adaptive_petrinets/reconfnet/-/tree/master/doc



more files from codiMD
Uwe Aßmann authored 9 months ago
06295069

Name	Last commit	Last update
..		
api	new readme in doc/api. 6 Modules are documented now.	11 months ago
Comments.txt	naturaldocs: new comment type Question, Algorithm.	11 months ago
Languages.txt	The language of GRSHELL was added in Languages.txt, to incl...	11 months ago
Makefile	naturaldocs: new comment type Question, Algorithm.	11 months ago
Project.txt	documented petri2dynamic nets	10 months ago
Readme.md	new readme in doc/api. 6 Modules are documented now.	11 months ago
protocol-reconfnet.md	more files from codiMD	9 months ago

Readme.md

Generation of API documentation for Reconfnet

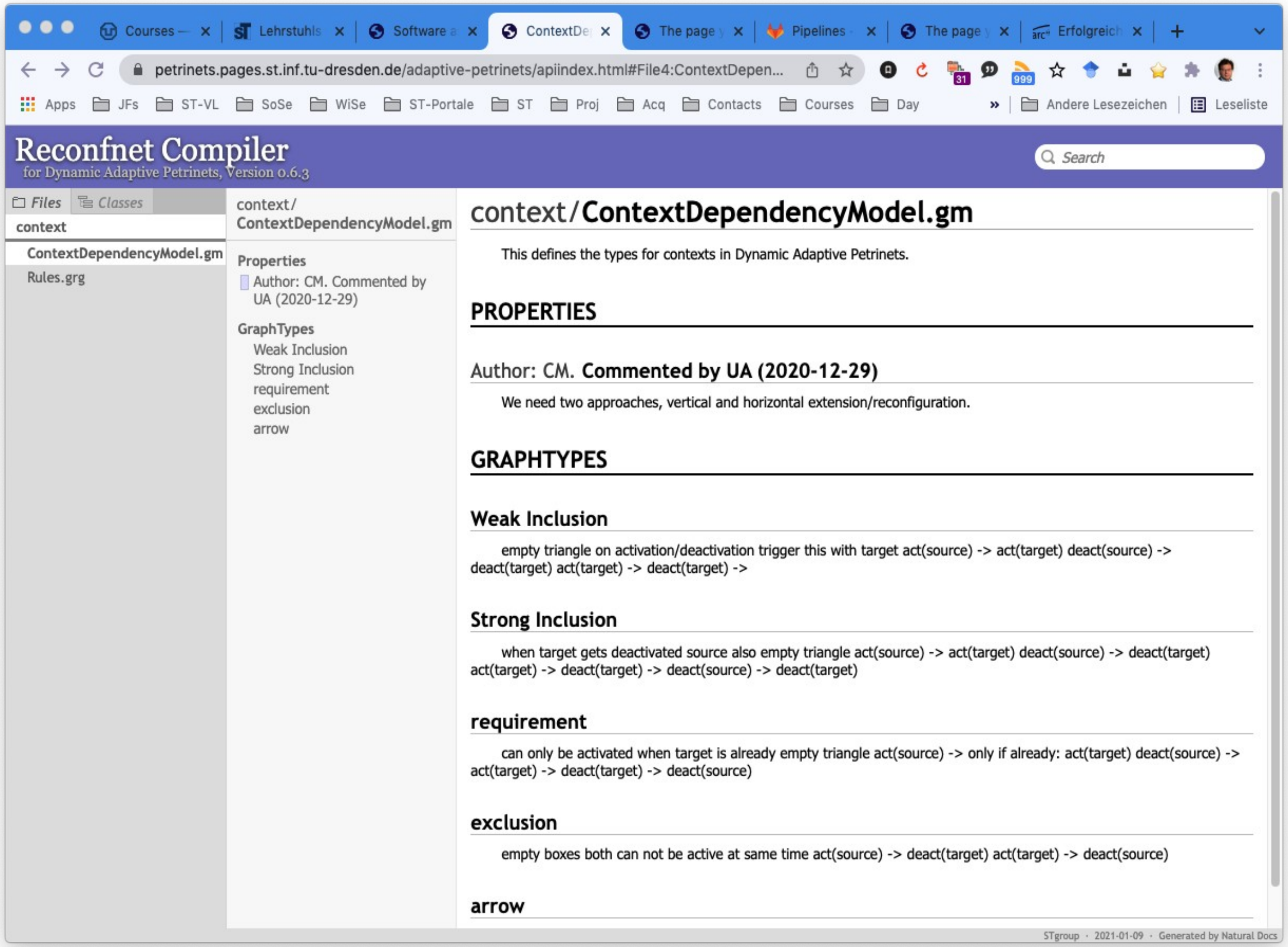
- Install naturaldocs (e.g., with brew).
- make sure that in your file system the projects **reconfnet** and its documentation project **reconfnet.pages.st.tu-dresden.de** are located as sister directories.
  - Naturaldocs needs to find the **relative paths** it is looking for.
  - At the moment naturaldocs is located in the **reconfnet/doc** directory; which might change to **reconfnet.pages.st.tu-dresden.de**, but this works only if relative paths are consistent.
- Then run:

```
cd doc
naturaldocs .
open api/index.html
```

- make sure that the subdirectory api is not deleted. naturaldocs relies on it

Example NaturalDocs API generated for GrGen

- GrGen.net is a generator for graph rewrite specifications (see Part IV)
- There is no specific API doc generator for GrGen, but NaturalDocs can be tailored to it



petrinets.pages.st.inf.tu-dresden.de/adaptive-petrinets/apiindex.html#File4:ContextDepen...
Reconfnet Compiler
for Dynamic Adaptive Petrinets, Version 0.6.3
Search
Files
Classes
context
ContextDependencyModel.gm
Rules.grg
context/
ContextDependencyModel.gm
Properties
Author: CM. Commented by UA (2020-12-29)
GraphTypes
Weak Inclusion
Strong Inclusion
requirement
exclusion
arrow

context/**ContextDependencyModel.gm**

This defines the types for contexts in Dynamic Adaptive Petrinets.

PROPERTIES

Author: CM. **Commented by UA (2020-12-29)**

We need two approaches, vertical and horizontal extension/reconfiguration.

GRAPHTYPES

Weak Inclusion

empty triangle on activation/deactivation trigger this with target act(source) -> act(target) deact(source) -> deact(target) act(target) -> deact(target) ->

Strong Inclusion

when target gets deactivated source also empty triangle act(source) -> act(target) deact(source) -> deact(target) act(target) -> deact(target) -> deact(source) -> deact(target)

requirement

can only be activated when target is already empty triangle act(source) -> only if already: act(target) deact(source) -> act(target) -> deact(target) -> deact(source)

exclusion

empty boxes both can not be active at same time act(source) -> deact(target) act(target) -> deact(source)

arrow

STgroup · 2021-01-09 · Generated by Natural Docs

The End

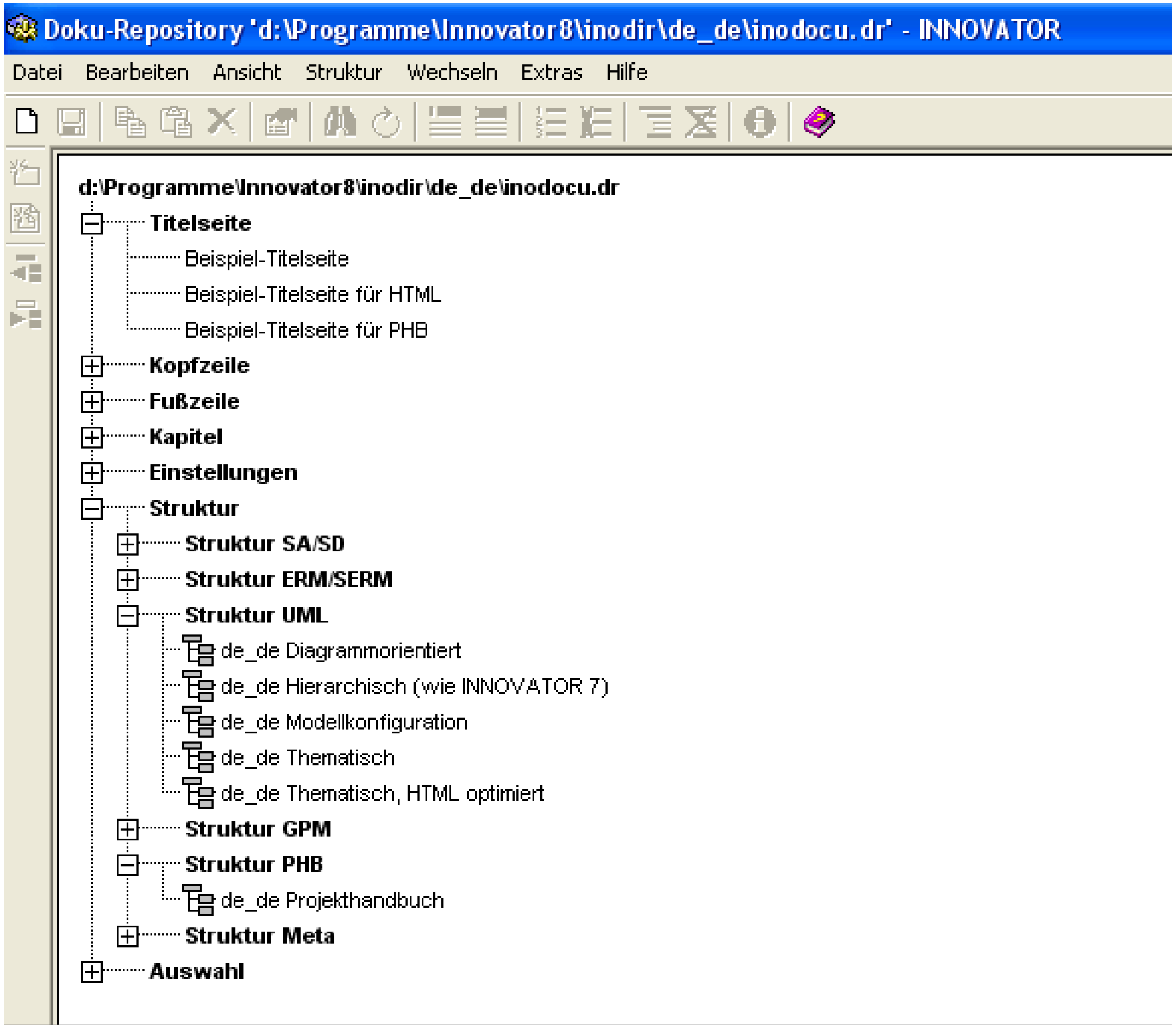
- Why is generation of documentation similar to code generation?
- Explain why a higher-order RAG is useful for documentation generation
- Which role does a pattern-matching language such as Xcerpt play in documentation generation?
- Why is the generation of documentation part of a macromodel?
- Why is a documentation a *derived model?*
- What happens if text from the API documentation flows back into the code as comments?

A.1 Other Template Expanders for Documentation Generation

Documentation Tools of MID Innovator

- Innovator provides documentation templates, into which diagrams, models, code can be embedded
- Several formats:
  - pdf
  - Word
  - ASCII
  - XML

Ex.: Innovator Documentation Template (Dokumentationsvorlage)

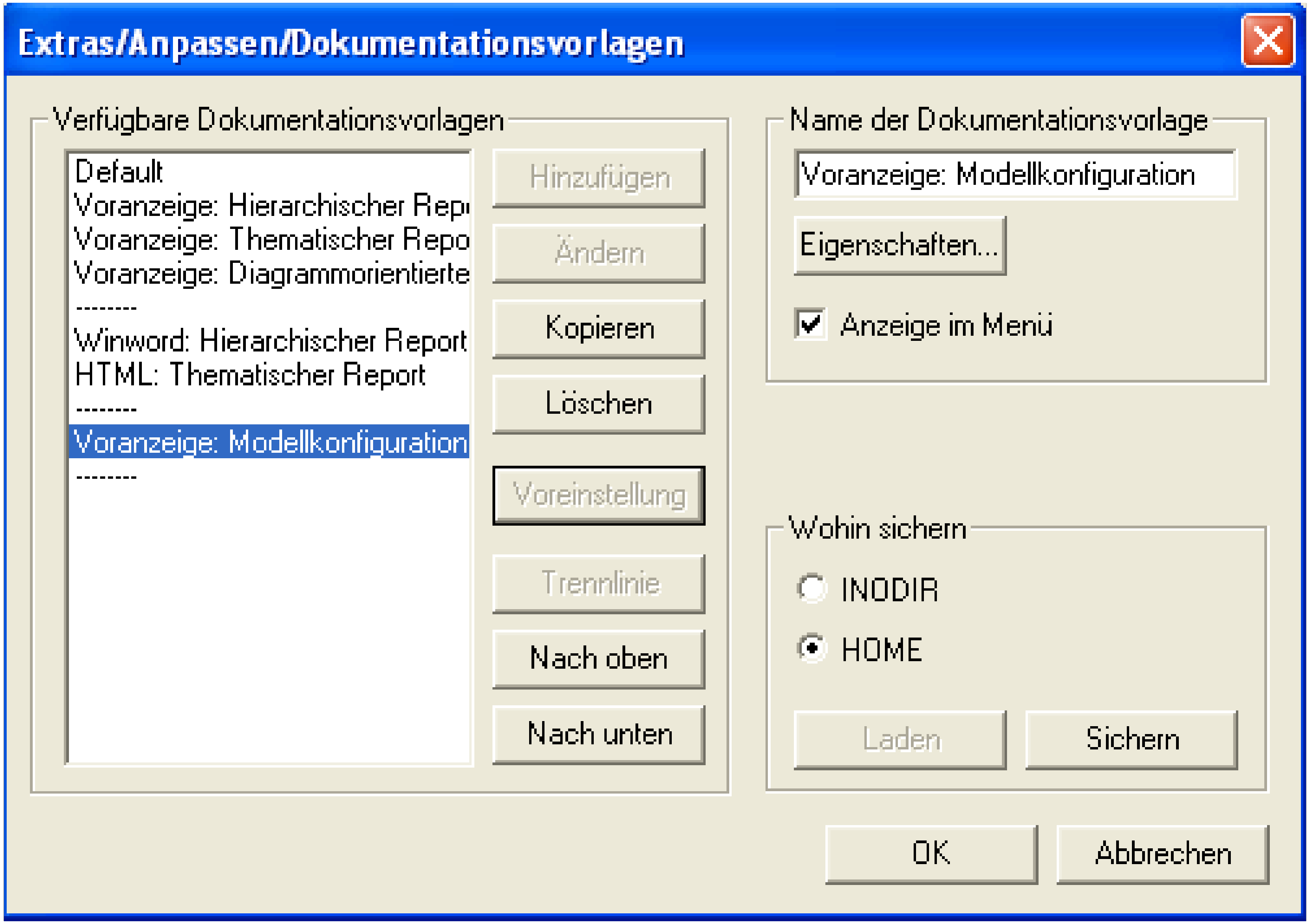


Doku-Repository 'd:\Programme\Innovator8\inodir\de_de\inodocu.dr' - INNOVATOR

Datei Bearbeiten Ansicht Struktur Wechseln Extras Hilfe

```
d:\Programme\Innovator8\inodir\de_de\inodocu.dr
⊟ Titelseite
    Beispiel-Titelseite
    Beispiel-Titelseite für HTML
    Beispiel-Titelseite für PHB
⊞ Kopfzeile
⊞ Fußzeile
⊞ Kapitel
⊞ Einstellungen
⊟ Struktur
    ⊞ Struktur SA/SD
    ⊞ Struktur ERM/SERM
    ⊟ Struktur UML
        de_de Diagrammorientiert
        de_de Hierarchisch (wie INNOVATOR 7)
        de_de Modellkonfiguration
        de_de Thematisch
        de_de Thematisch, HTML optimiert
    ⊞ Struktur GPM
    ⊟ Struktur PHB
        de_de Projekthandbuch
    ⊞ Struktur Meta
⊞ Auswahl
```

Ex.: Innovator Documentation Template (Dokumentationsvorlage): Adaptation



Extras/Anpassen/Dokumentationsvorlagen

Verfügbare Dokumentationsvorlagen

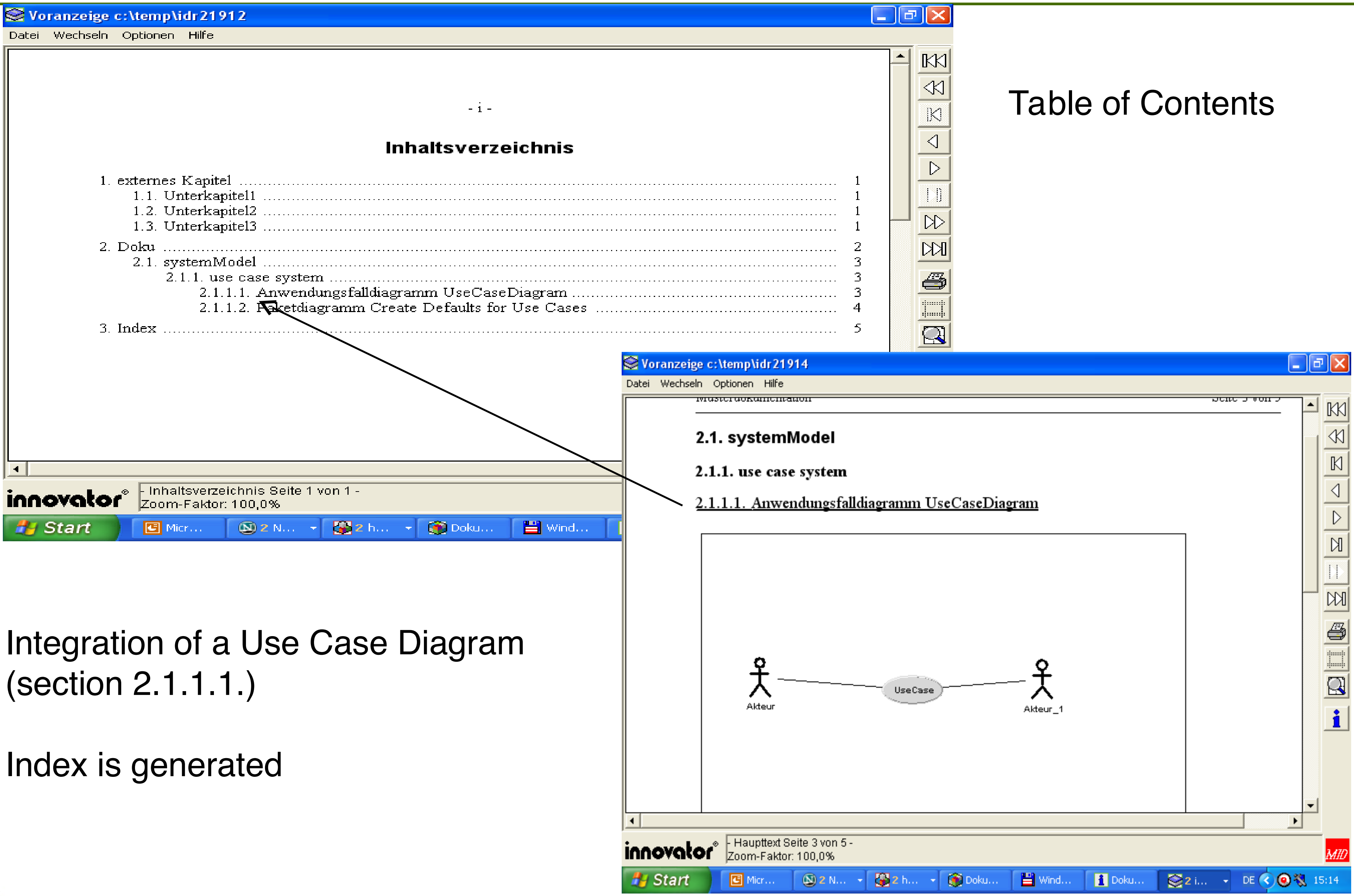
- Default
- Voranzeige: Hierarchischer Rep
- Voranzeige: Thematischer Repo
- Voranzeige: Diagrammorientierte
- --------
- Winword: Hierarchischer Report
- HTML: Thematischer Report
- --------
- Voranzeige: Modellkonfiguration
- --------

Name der Dokumentationsvorlage: Voranzeige: Modellkonfiguration

[✓] Anzeige im Menü

Wohin sichern: INODIR [✓ HOME]

Innovator - Generated Example Word Document


Voranzeige c:\temp\idr21912
Datei Wechseln Optionen Hilfe

- i -

Inhaltsverzeichnis

1. externes Kapitel 1
 1.1. Unterkapitel1 1
 1.2. Unterkapitel2 1
 1.3. Unterkapitel3 1
2. Doku 2
 2.1. systemModel 3
  2.1.1. use case system 3
   2.1.1.1. Anwendungsfalldiagramm UseCaseDiagram 3
   2.1.1.2. Paketdiagramm Create Defaults for Use Cases 4
3. Index 5

innovator
- Inhaltsverzeichnis Seite 1 von 1 -
Zoom-Faktor: 100,0%
Start
Micr...
2 N...
2 h...
Doku...
Wind...
Voranzeige c:\temp\idr21914
Datei Wechseln Optionen Hilfe

Musterdokumentation Seite 3 von 5

2.1. systemModel

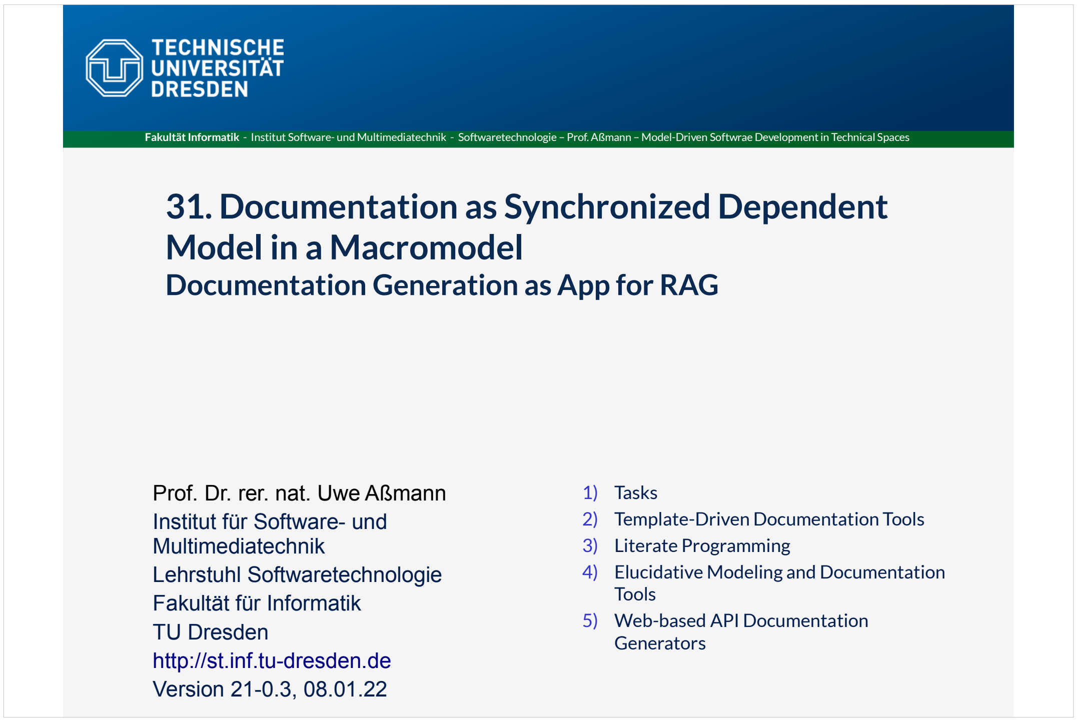
2.1.1. use case system

2.1.1.1. Anwendungsfalldiagramm UseCaseDiagram

Table of Contents

Integration of a Use Case Diagram (section 2.1.1.1.)

Index is generated



31. Documentation as Synchronized Dependent Model in a Macromodel

Documentation Generation as App for RAG

Prof. Dr. rer. nat. Uwe Aßmann
Institut für Software- und Multimediatechnik
Lehrstuhl Softwaretechnologie
Fakultät für Informatik
TU Dresden
http://st.inf.tu-dresden.de

Version 21-0.3, 08.01.22

1) Tasks
2) Template-Driven Documentation Tools
3) Literate Programming
4) Elucidative Modeling and Documentation Tools
5) Web-based API Documentation Generators

mehr code examples mit xcerpt und EMod

References

- D. E. Knuth, Literate Programming, The Computer Journal, Volume 27, Issue 2, 1984, Pages 97–111, https://doi.org/10.1093/comjnl/27.2.97
- D. Cordes and M. Brown, "The literate-programming paradigm," in Computer, vol. 24, no. 6, pp. 52-61, June 1991, doi: 10.1109/2.86838.
- Kurt Nørmark. Elucidative programming. Nordic Journal of Computing, 2000. Citeseer: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.408.2506&rep=rep1&type=pdf

- C. Wilke, A. Bartho, J. Schroeter, S. Karol, U. Aßmann. Elucidative Development for Model-Based Documentation and Language Specification (Extended Version). Technische Universität Dresden. Institut fur Software- und Multimediatechnik. Technical Reports TUD-FI12-01-Januar 2012, ISSN 1430-211X.
  - http://nbn-resolving.de/urn:nbn:de:bsz:14-qucosa-83442
- Andreas Bartho. Elucidative Modeling. PhD thesis, Technische Universität Dresden, Fakultät Informatik, May 2014.
  - http://nbn-resolving.de/urn:nbn:de:bsz:14-qucosa-208060
  - https://www.linkedin.com/pub/andreas-bartho/ba/922/8a4?trk=pub-pbmap

Interesting

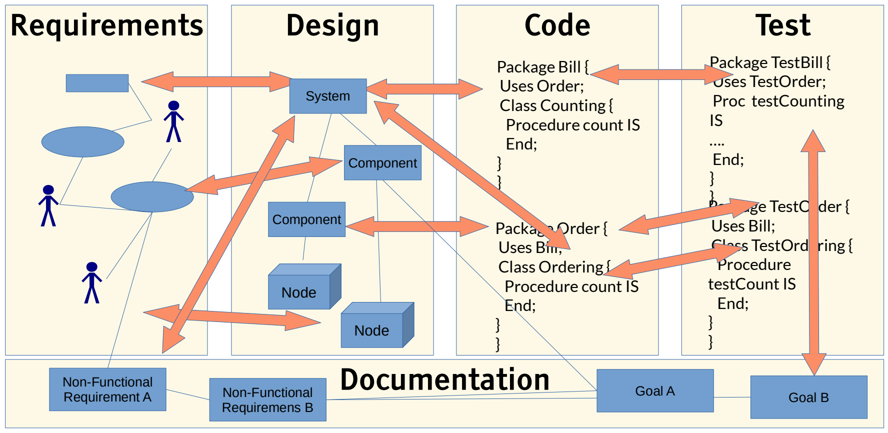
- https://www.writethedocs.org/ is a conference for documentation practicioneers
- https://waset.org/software-implementation-and-software-documentation-conference

31.1 Tasks of Documentation Tools

http://en.wikipedia.org/wiki/Software_documentation

Q12: The ReDoDeCT Problem and its Macromodel

- The **ReDoDeCT problem** is the problem how requirements, documentation, design, code, and tests are related (→ V model)
- Mappings between the Requirements model, Documentation files, Design model, Code, Test cases
- A **ReDoDeCT macromodel** has maintained mappings between all 5 models



Basics of Software Documentation

- Documentation is a means of **communication** to keep software alive
  - between developers and future developers
  - between coders and testers
  - between developers and managers (for reviews and audits)
- Problems:
  - Documentation ***ages*** because code is modified and evolved (***documentation aging)***
  - Good documentation costs time and money
- Different kinds of documentation:
  - **Generated documentation** is derived from code and models
  - **Integrated Documentation** is derived from the code (e.g., in comments), e.g., JavaDoc
  - **Elucidative Documentation**, derives both from another and keeps it consistent (generative or round-trip engineering)
- Standards:
  - national DIN 66230, 66231, 66232, 66270(1998)
  - international ISO/IEC 6592(2000), ISO/IEC 18019(2004)

Without documentation, a program is not software

Quelle: [24 S. 241 ff.]

Taxonomy of Documentation Documents

- **User documentation** (Benutzerdokumentation) explains the program to end users
  - Tutorials, user handbook, online documentation
- **System documentation** for installation, test cases, code documentation, maintenance, operations
  - **API documentation** documents interfaces of the system or framework, to let programmers use them for writing apps
  - **Architecture documentation** to highlight the architectural structure of the software, e.t., with arc42 (https://www.arc42.de/)
- **Project documentation**
  - Developer documentation
  - Project documentation (project plan, requriements specification, status reports, after study)
- **Quality documentation**
  - Test-, review, audit documentation
- **Process documentation**
  - Standards, processes

Quelle: [24 S. 245 ff.]

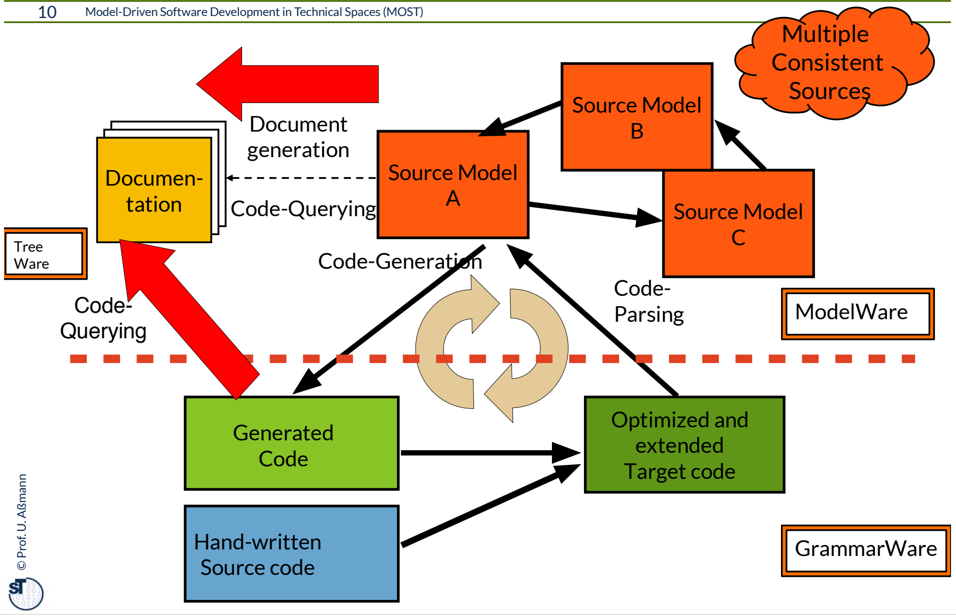
Tasks of Documentation Tools

- Basically, documentation generation is similar to code generation. Documentation is created in higher-order attributes on a link tree by a RAG
- **Documentation generation is an application areas for RAG**
- **Generation** of derived documents from code and models
  - Generation of Word (docx), LibreOffice (odt), rtf, xml, html formats
  - Generation of figures (svg, png, pdf)
  - Generation of snippets and generic snippets
  - Back-linking to originals
- **Filling** of documentation templates (with the hedge-principle)
- **Parameterization** with layouts
  - via css-style sheets

31.2 Generative, Template-Driven Documentation Tools

.. Documentation derived from code and models, based on template-based code generation

Macromodel Principle and Round-Trip Engineering



Documentation Tool JavaDoc is a Template Expander

- JavaDoc reads Java source code and extracts html from the code comments, based on **html templates**
  - Typical hedge-based code generation with generic snippets
- Generation of additional contents and indices
- Controlled by Java metadata attributes
  - @author, @date, @param
- Layouting via plugin classes called *doclets*
- JavaDoc has been realized for all programming languages

[JavaDoc; Wikipedia: http://de.wikipedia.org/wiki/JavaDoc]

JavaDoc is a Typical HRAG Application

- The html documentation is computed in a higher-order synthesized attribute `htmldoc:HTML`

```
// schematic, synthesis from bottom to top
Interpretation javaDoc(Tree → Tree)  {
  Attributions of Root(classes[]) {
    this.htmldoc := map + classes.htmldoc;
    <println(„Result is %S“, this.htmldoc)>
  }
  Attributions of Class(superclass:Class,methods[]) {
    this.htmldoc := <superclass.Name + methods.htmldoc;
  }
  Attributions of Method(name,comment) {
    this.htmldoc := „<h1>“+name+“</h2>“+comment.htmldoc;
  }
  Attributions of Comment(text) {
    this.htmldoc := text;
  }
}
```

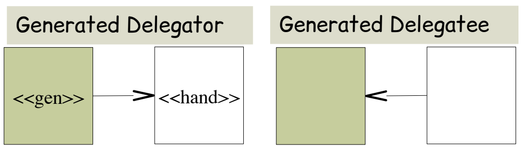
Composition of Separated Hand-Written and Generated Documentation Snippets

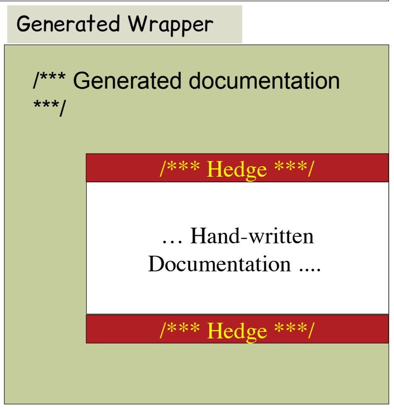
In separate files: Coupling by "include"

- Only possible if document format supports subdocument inclusion
  - e.g., TeX or Framemaker

In one file:

Coupling with **hegdes (Trennmarkierung)**

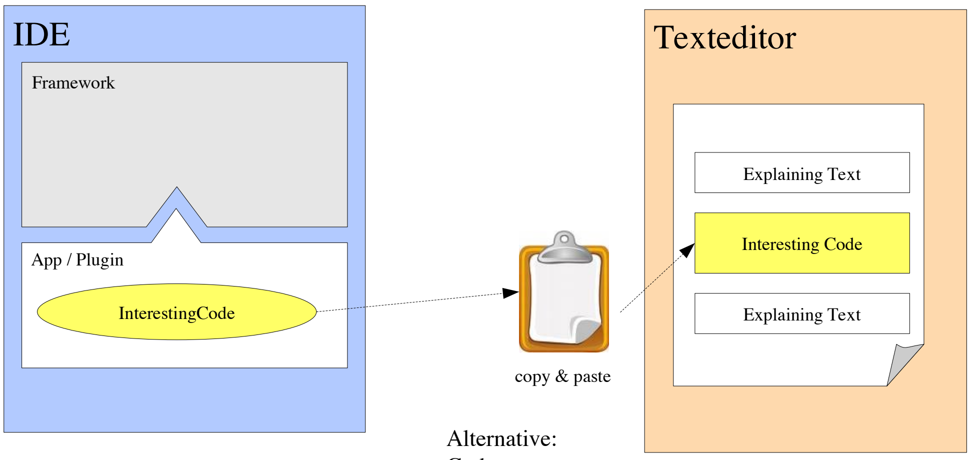



Generated Wrapper
/*** Generated documentation ***/
/*** Hedge ***/
… Hand-written Documentation ....
/*** Hedge ***/

31.3 Literate Programming

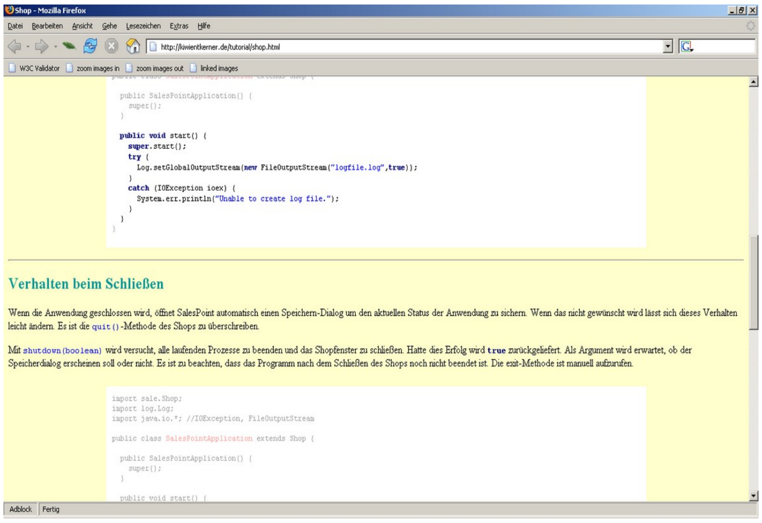
- They integrate code, models and documentation by **separating code from documentation**

Classic: Manual Writing of Tutorials



Alternative:
Code query
e.g., with
Xcerpt or QL

How to Write Integrated Documentation and Tutorials?


Shop - Mozilla Firefox
Datei Bearbeiten Ansicht Gehe Lesezeichen Extras Hilfe
http://kiwientkerner.de/tutorial/shop.html
W3C Validator
zoom images in
zoom images out
linked images
Adblock
Fertig

```
  public SalesPointApplication() {
    super();
  }

  public void start() {
    super.start();
    try {
      Log.setGlobalOutputStream(new FileOutputStream("logfile.log",true));
    }
    catch (IOException ioex) {
      System.err.println("Unable to create log file.");
    }
  }
}
```

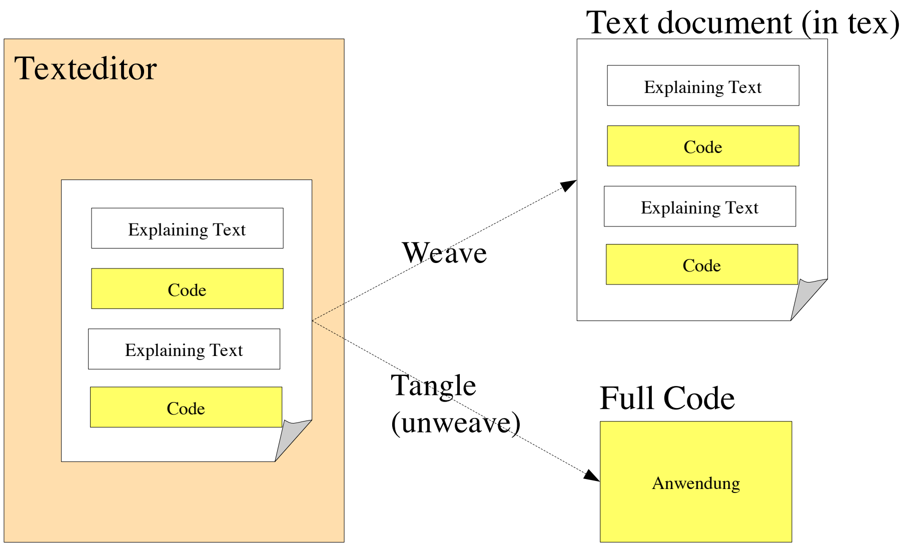
Verhalten beim Schließen

Wenn die Anwendung geschlossen wird, öffnet SalesPoint automatisch einen Speichern-Dialog um den aktuellen Status der Anwendung zu sichern. Wenn das nicht gewünscht wird lässt sich dieses Verhalten leicht ändern. Es ist die `quit()`-Methode des Shops zu überschreiben.

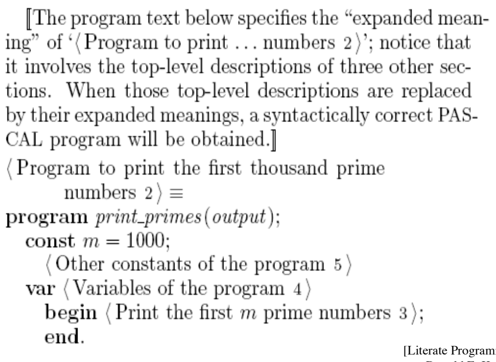
Mit `shutdown(boolean)` wird versucht, alle laufenden Prozesse zu beenden und das Shopfenster zu schließen. Hatte dies Erfolg wird `true` zurückgeliefert. Als Argument wird erwartet, ob der Speicherdialog erscheinen soll oder nicht. Es ist zu beachten, dass das Programm nach dem Schließen des Shops noch nicht beendet ist. Die exit-Methode ist manuell aufzurufen.

```
import sale.Shop;
import log.Log;
import java.io.*; //IOException, FileOutputStream

public class SalesPointApplication extends Shop {

  public SalesPointApplication() {
    super();
  }

  public void start() {
```

[Knuth] Literate Programming by Code Unweaving



Literate Programming



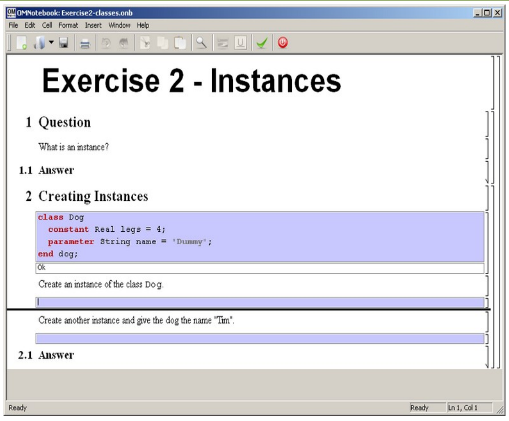
⟦The program text below specifies the "expanded meaning" of '⟨Program to print … numbers 2⟩'; notice that it involves the top-level descriptions of three other sections. When those top-level descriptions are replaced by their expanded meanings, a syntactically correct PASCAL program will be obtained.⟧

```
⟨Program to print the first thousand prime
        numbers 2⟩ ≡
program print_primes(output);
  const m = 1000;
    ⟨Other constants of the program 5⟩
  var ⟨Variables of the program 4⟩
    begin ⟨Print the first m prime numbers 3⟩;
    end.
```

[Literate Programming
von Donald E. Knuth]

- The TeX engine is programmed literately
- Overview: http://www.literateprogramming.com/
- OMNotebook/DrModelica: http://www.modelica.org/tools

OMNotebook – Literate Programming with DrModelica


OMNotebook: Exercise2-classes.onb
File Edit Cell Format Insert Window Help

Exercise 2 - Instances

1 Question

What is an instance?

1.1 Answer

2 Creating Instances

```
class Dog
  constant Real legs = 4;
  parameter String name = "Dummy";
end dog;
```

```
Ok
```

Create an instance of the class Dog.

Create another instance and give the dog the name "Tim".

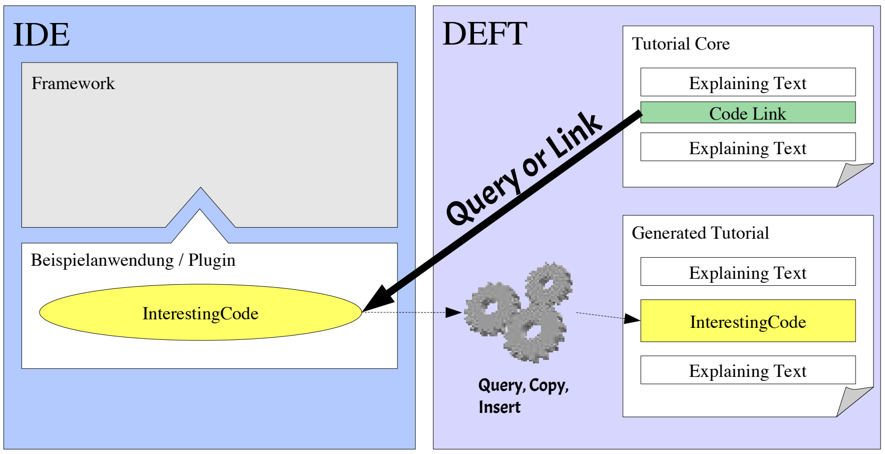
2.1 Answer

- Linked documents with interactive exercises
- Inspired by DrScheme und DrJava, learning tools for Scheme resp. Java
- www.openmodelica.org

31.4 Elucidative Documentation Tools

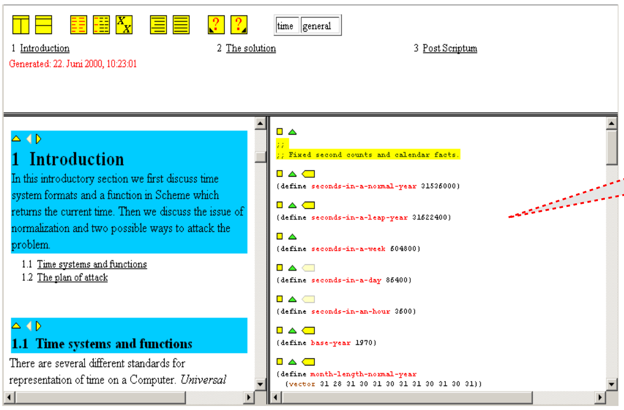
- They link code, models and documentation by **model and code mapping**
- **and renew the documentation by *hot updates***

Elucidative Programming Links Documentation with Queries to Code



hot update
(hot synchronisation):
round-trip engineering

Elucidative Programming



time general

1 Introduction 2 The solution 3 Post Scriptum

Generated: 22. Juni 2000, 10:23:01

1 Introduction

In this introductory section we first discuss time system formats and a function in Scheme which returns the current time. Then we discuss the issue of normalization and two possible ways to attack the problem.

1.1 Time systems and functions
1.2 The plan of attack

1.1 Time systems and functions

There are several different standards for representation of time on a Computer. *Universal*

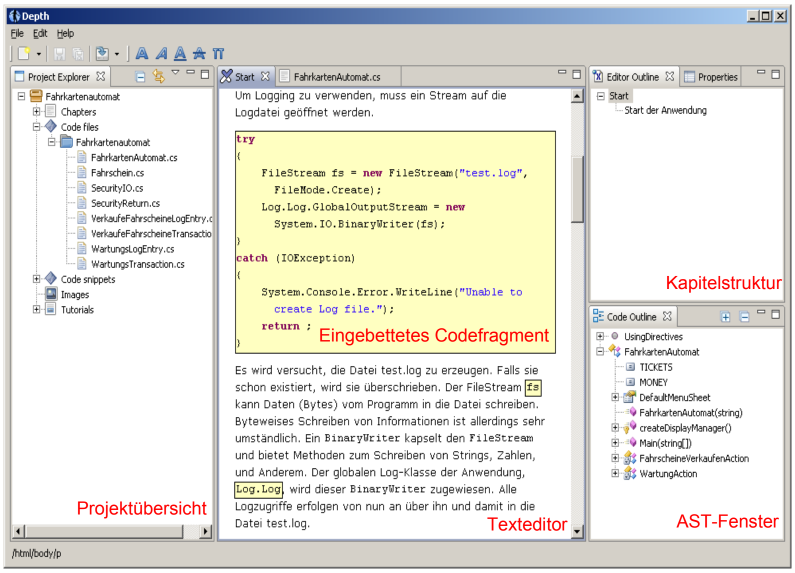
```
;;
;; Fixed second counts and calendar facts.

(define seconds-in-a-normal-year 31536000)

(define seconds-in-a-leap-year 31622400)

(define seconds-in-a-week 604800)

(define seconds-in-a-day 86400)

(define seconds-in-an-hour 3600)

(define base-year 1970)

(define month-length-normal-year
  (vector 31 28 31 30 31 30 31 31 30 31 30 31))
```

„Scheme Elucidator" Environment

- Elucidative Programming shows documentation and code in parallel
- http://www.cs.aau.dk/~normark/elucidative-programming/
- http://deftproject.org

hot update
(hot synchronisation):
round-trip engineering

Development Environment For Tutorials (DEFT www.deftproject.org)

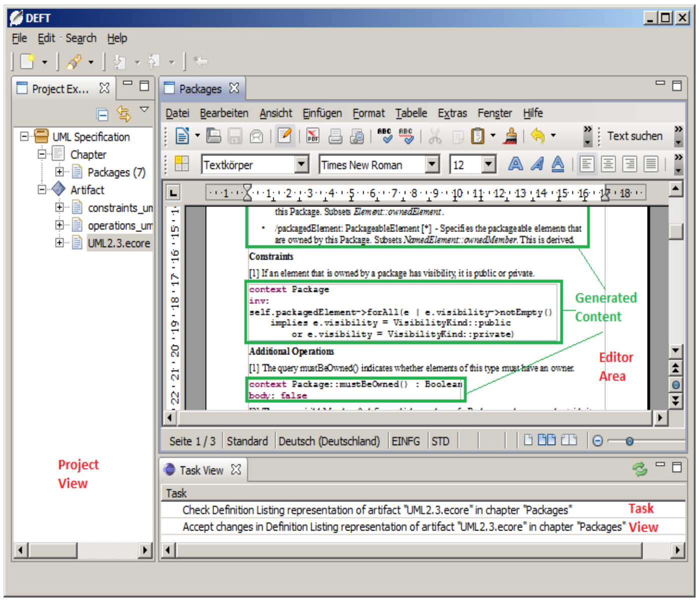


Um Logging zu verwenden, muss ein Stream auf die Logdatei geöffnet werden.

```
try
{
    FileStream fs = new FileStream("test.log",
      FileMode.Create);
    Log.Log.GlobalOutputStream = new
      System.IO.BinaryWriter(fs);
}
catch (IOException)
{
    System.Console.Error.WriteLine("Unable to
      create Log file.");
    return ;
}
```

Es wird versucht, die Datei test.log zu erzeugen. Falls sie schon existiert, wird sie überschrieben. Der FileStream `fs` kann Daten (Bytes) vom Programm in die Datei schreiben. Byteweises Schreiben von Informationen ist allerdings sehr umständlich. Ein `BinaryWriter` kapselt den `FileStream` und bietet Methoden zum Schreiben von Strings, Zahlen, und Anderem. Der globalen Log-Klasse der Anwendung, `Log.Log`, wird dieser `BinaryWriter` zugewiesen. Alle Logzugriffe erfolgen von nun an über ihn und damit in die Datei test.log.

Embedding UML Constraints for UML Models into Documentation



this Package. Subsets *Element::ownedElement*.

- /packagedElement: PackageableElement [*] - Specifies the packageable elements that are owned by this Package. Subsets *NamedElement::ownedMember*. This is derived.

Constraints

[1] If an element that is owned by a package has visibility, it is public or private.

```
context Package
inv:
self.packagedElement->forAll(e | e.visibility->notEmpty()
   implies e.visibility = VisibilityKind::public
      or e.visibility = VisibilityKind::private)
```

Additional Operations

[1] The query mustBeOwned() indicates whether elements of this type must have an owner.

```
context Package::mustBeOwned() : Boolean
body: false
```

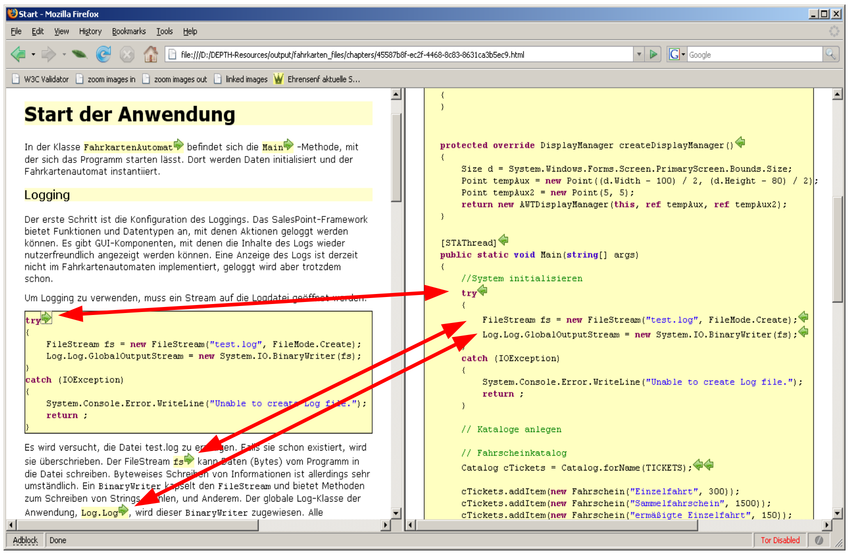
Check Definition Listing representation of artifact "UML2.3.ecore" in chapter "Packages"

Accept changes in Definition Listing representation of artifact "UML2.3.ecore" in chapter "Packages"

Development Environment For Tutorials (DEFT)

- Eclipse RCP application, language independent
- Management of code, models and text
- Prettyprinting of code fragments from code templates
- Hot update of generated documentation
    - Automatic update of embedded code fragments
    - Notiviation if code fragments have changed

Generated HTML Tutorial


Start - Mozilla Firefox
File Edit View History Bookmarks Tools Help
file:///D:/DEPTH-Resources/output/fahrkarten_files/chapters/45587b8f-ec2f-4468-8c83-8631ca3b5ec9.html
Google
W3C Validator
zoom images in
zoom images out
linked images
Ehrensenf aktuelle S...
Adblock
Done
Tor Disabled

Start der Anwendung

In der Klasse `FahrkartenAutomat` befindet sich die `Main` -Methode, mit der sich das Programm starten lässt. Dort werden Daten initialisiert und der Fahrkartenautomat instantiiert.

Logging

Der erste Schritt ist die Konfiguration des Loggings. Das SalesPoint-Framework bietet Funktionen und Datentypen an, mit denen Aktionen geloggt werden können. Es gibt GUI-Komponenten, mit denen die Inhalte des Logs wieder nutzerfreundlich angezeigt werden können. Eine Anzeige des Logs ist derzeit nicht im Fahrkartenautomaten implementiert, geloggt wird aber trotzdem schon.

Um Logging zu verwenden, muss ein Stream auf die Logdatei geöffnet werden.

```
try
{
    FileStream fs = new FileStream("test.log", FileMode.Create);
    Log.Log.GlobalOutputStream = new System.IO.BinaryWriter(fs);
}
catch (IOException)
{
    System.Console.Error.WriteLine("Unable to create Log file.");
    return ;
}
```

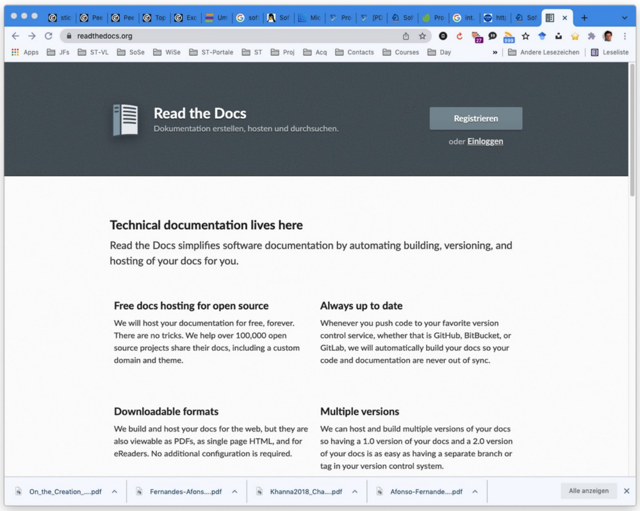
Es wird versucht, die Datei test.log zu erzeugen. Falls sie schon existiert, wird sie überschrieben. Der FileStream `fs` kann Daten (Bytes) vom Programm in die Datei schreiben. Byteweises Schreiben von Informationen ist allerdings sehr umständlich. Ein `BinaryWriter` kapselt den `FileStream` und bietet Methoden zum Schreiben von Strings, Zahlen, und Anderem. Der globale Log-Klasse der Anwendung, `Log.Log`, wird dieser `BinaryWriter` zugewiesen. Alle

```
    {
    }


    protected override DisplayManager createDisplayManager()
    {
        Size d = System.Windows.Forms.Screen.PrimaryScreen.Bounds.Size;
        Point tempAux = new Point((d.Width - 100) / 2, (d.Height - 80) / 2);
        Point tempAux2 = new Point(5, 5);
        return new AWTDisplayManager(this, ref tempAux, ref tempAux2);
    }

    [STAThread]
    public static void Main(string[] args)
    {
        //System initialisieren
        try
        {
            FileStream fs = new FileStream("test.log", FileMode.Create);
            Log.Log.GlobalOutputStream = new System.IO.BinaryWriter(fs);
        }
        catch (IOException)
        {
            System.Console.Error.WriteLine("Unable to create Log file.");
            return ;
        }

        // Kataloge anlegen

        // Fahrscheinkatalog
        Catalog cTickets = Catalog.forName(TICKETS);

        cTickets.addItem(new Fahrschein("Einzelfahrt", 300));
        cTickets.addItem(new Fahrschein("Sammelfahrschein", 1500));
        cTickets.addItem(new Fahrschein("ermäßigte Einzelfahrt", 150));
```

31.5 Web-based Documentation Generators based on Markdown

Sphinx and the Documentation Cloud readthedocs.org

- readthedocs is a cloud for documentation projects
- supporting two documentation generators *sphinx* and *mkdocs*



Read the Docs
Dokumentation erstellen, hosten und durchsuchen.

Technical documentation lives here

Read the Docs simplifies software documentation by automating building, versioning, and hosting of your docs for you.

Free docs hosting for open source

We will host your documentation for free, forever. There are no tricks. We help over 100,000 open source projects share their docs, including a custom domain and theme.

Always up to date

Whenever you push code to your favorite version control service, whether that is GitHub, BitBucket, or GitLab, we will automatically build your docs so your code and documentation are never out of sync.

Downloadable formats

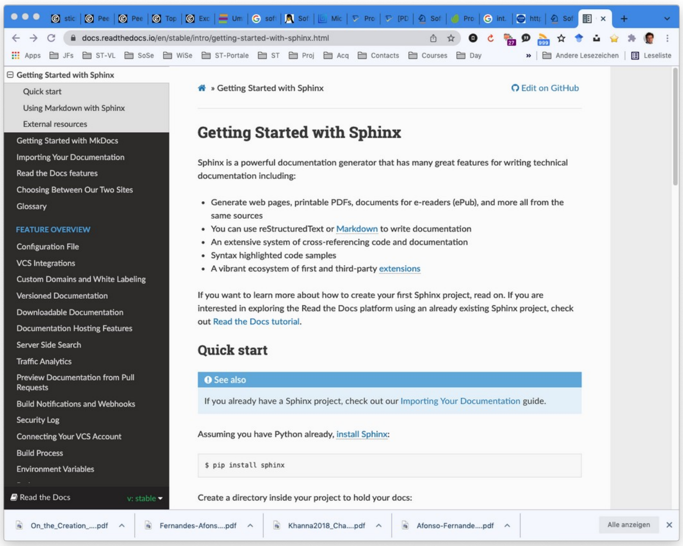
We build and host your docs for the web, but they are also viewable as PDFs, as single page HTML, and for eReaders. No additional configuration is required.

Multiple versions

We can host and build multiple versions of your docs so having a 1.0 version of your docs and a 2.0 version of your docs is as easy as having a separate branch or tag in your version control system.

Sphinx

- Architecture documentation
- User documentation
- Files in formats restructuredText and Markdown are transformed to HTML
- Treats entire directories
- many output formats (e.g., Latex)
- Can be coupled with Javadoc or similar API doc generators



Getting Started with Sphinx

Sphinx is a powerful documentation generator that has many great features for writing technical documentation including:

- Generate web pages, printable PDFs, documents for e-readers (ePub), and more all from the same sources
- You can use reStructuredText or Markdown to write documentation
- An extensive system of cross-referencing code and documentation
- Syntax highlighted code samples
- A vibrant ecosystem of first and third-party extensions

If you want to learn more about how to create your first Sphinx project, read on. If you are interested in exploring the Read the Docs platform using an already existing Sphinx project, check out Read the Docs tutorial.

Quick start

See also

If you already have a Sphinx project, check out our Importing Your Documentation guide.

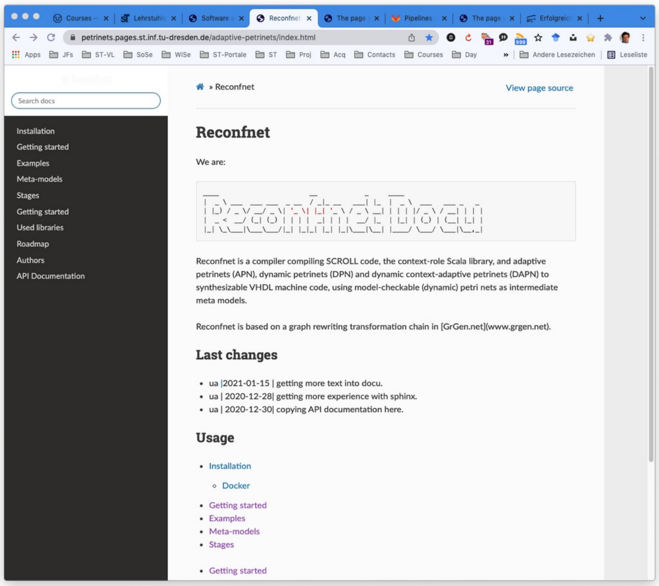
Assuming you have Python already, install Sphinx:

```
$ pip install sphinx
```

Create a directory inside your project to hold your docs:

Example Sphinx Project

- Petrinet compiler Reconfnet https://petrinets.pages.st.inf.tu-dresden.de/adaptive-petrinets/index.html
-



Reconfnet

We are:

Reconfnet is a compiler compiling SCROLL code, the context-role Scala library, and adaptive petrinets (APN), dynamic petrinets (DPN) and dynamic context-adaptive petrinets (DAPN) to synthesizable VHDL machine code, using model-checkable (dynamic) petri nets as intermediate meta models.

Reconfnet is based on a graph rewriting transformation chain in [GrGen.net](www.grgen.net).

Last changes

- ua |2021-01-15 | getting more text into docu.
- ua | 2020-12-28| getting more experience with sphinx.
- ua | 2020-12-30| copying API documentation here.

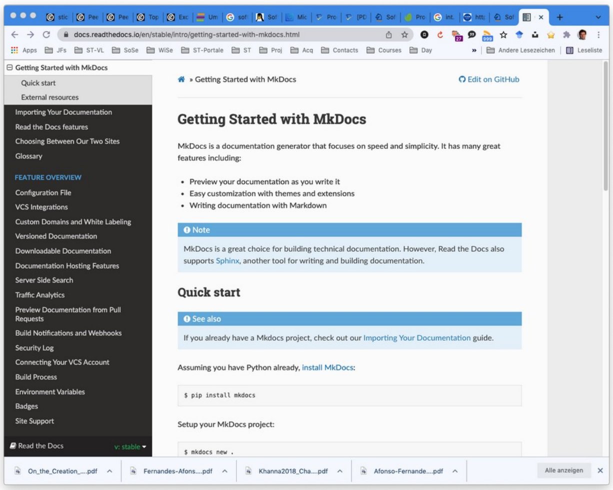
Usage

- Installation
  - Docker
- Getting started
- Examples
- Meta-models
- Stages

- Getting started

MkDocs

- Markdown files to HTML files
- several output formats



docs.readthedocs.io/en/stable/intro/getting-started-with-mkdocs.html
Apps JFs ST-VL SoSe WiSe ST-Portale ST Proj Acq Contacts Courses Day Andere Lesezeichen Leseliste
Getting Started with MkDocs
Quick start
External resources
Importing Your Documentation
Read the Docs features
Choosing Between Our Two Sites
Glossary
FEATURE OVERVIEW
Configuration File
VCS Integrations
Custom Domains and White Labeling
Versioned Documentation
Downloadable Documentation
Documentation Hosting Features
Server Side Search
Traffic Analytics
Preview Documentation from Pull Requests
Build Notifications and Webhooks
Security Log
Connecting Your VCS Account
Build Process
Environment Variables
Badges
Site Support
Read the Docs
v: stable
» Getting Started with MkDocs
Edit on GitHub
On_the_Creation_....pdf
Fernandes-Afons....pdf
Khanna2018_Cha....pdf
Afonso-Fernande....pdf
Alle anzeigen

Getting Started with MkDocs

MkDocs is a documentation generator that focuses on speed and simplicity. It has many great features including:

- Preview your documentation as you write it
- Easy customization with themes and extensions
- Writing documentation with Markdown

Note

MkDocs is a great choice for building technical documentation. However, Read the Docs also supports Sphinx, another tool for writing and building documentation.

Quick start

See also

If you already have a Mkdocs project, check out our Importing Your Documentation guide.

Assuming you have Python already, install MkDocs:

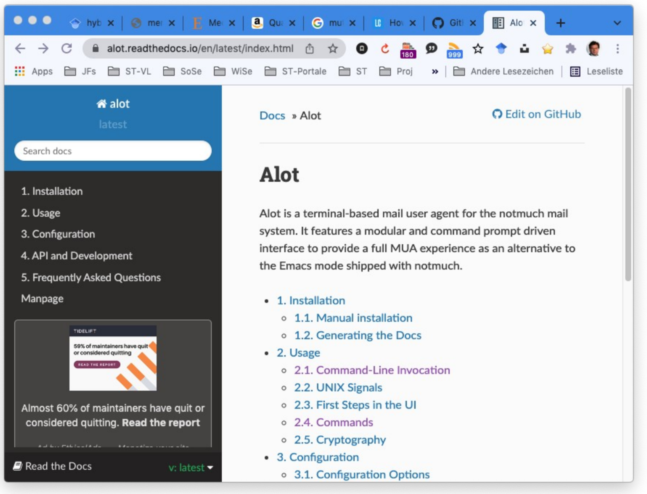
```
$ pip install mkdocs
```

Setup your MkDocs project:

```
$ mkdocs new .
```

Alot – a Mail User Agent Documented on readthedocs

- https://alot.readthedocs.io/


alot.readthedocs.io/en/latest/index.html
alot
latest
Search docs
1. Installation
2. Usage
3. Configuration
4. API and Development
5. Frequently Asked Questions
Manpage
Almost 60% of maintainers have quit or considered quitting. Read the report
Read the Docs
v: latest
Docs » Alot
Edit on GitHub

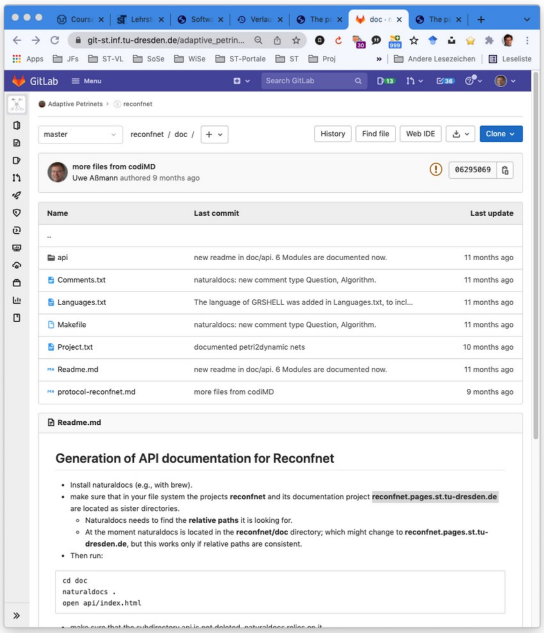
Alot

Alot is a terminal-based mail user agent for the notmuch mail system. It features a modular and command prompt driven interface to provide a full MUA experience as an alternative to the Emacs mode shipped with notmuch.

- 1. Installation
  - 1.1. Manual installation
  - 1.2. Generating the Docs
- 2. Usage
  - 2.1. Command-Line Invocation
  - 2.2. UNIX Signals
  - 2.3. First Steps in the UI
  - 2.4. Commands
  - 2.5. Cryptography
- 3. Configuration
  - 3.1. Configuration Options

NaturalDocs Generic API Documentation Generator

- Similar to JavaDoc, but more than 20 languages
- own keywords can be defined
- Example gitlab project from which API documentation for GrGen can be generated
  - https://git-st.inf.tu-dresden.de/adaptive_petrinets/reconfnet/-/tree/master/doc



more files from codiMD
Uwe Aßmann authored 9 months ago
06295069

Name	Last commit	Last update
..		
api	new readme in doc/api. 6 Modules are documented now.	11 months ago
Comments.txt	naturaldocs: new comment type Question, Algorithm.	11 months ago
Languages.txt	The language of GRSHELL was added in Languages.txt, to incl...	11 months ago
Makefile	naturaldocs: new comment type Question, Algorithm.	11 months ago
Project.txt	documented petri2dynamic nets	10 months ago
Readme.md	new readme in doc/api. 6 Modules are documented now.	11 months ago
protocol-reconfnet.md	more files from codiMD	9 months ago

Readme.md

Generation of API documentation for Reconfnet

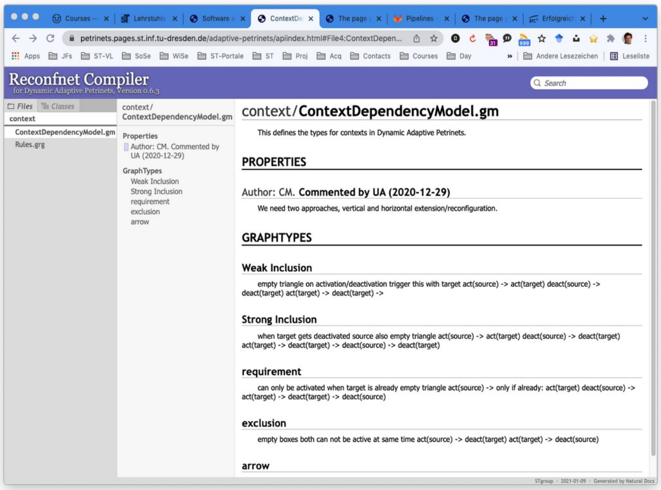
- Install naturaldocs (e.g., with brew).
- make sure that in your file system the projects **reconfnet** and its documentation project **reconfnet.pages.st.tu-dresden.de** are located as sister directories.
  - Naturaldocs needs to find the **relative paths** it is looking for.
  - At the moment naturaldocs is located in the **reconfnet/doc** directory; which might change to **reconfnet.pages.st.tu-dresden.de**, but this works only if relative paths are consistent.
- Then run:

```
cd doc
naturaldocs .
open api/index.html
```

- make sure that the subdirectory api is not deleted, naturaldocs relies on it

Example NaturalDocs API generated for GrGen

- GrGen.net is a generator for graph rewrite specifications (see Part IV)
- There is no specific API doc generator for GrGen, but NaturalDocs can be tailored to it



Reconfnet Compiler
for Dynamic Adaptive Petrinets, Version 0.6.3

context/**ContextDependencyModel.gm**

This defines the types for contexts in Dynamic Adaptive Petrinets.

PROPERTIES

Author: CM. **Commented by UA (2020-12-29)**

We need two approaches, vertical and horizontal extension/reconfiguration.

GRAPHTYPES

Weak Inclusion

empty triangle on activation/deactivation trigger this with target act(source) -> act(target) deact(source) -> deact(target) act(target) -> deact(target) ->

Strong Inclusion

when target gets deactivated source also empty triangle act(source) -> act(target) deact(source) -> deact(target) act(target) -> deact(target) -> deact(source) -> deact(target)

requirement

can only be activated when target is already empty triangle act(source) -> only if already: act(target) deact(source) -> act(target) -> deact(target) -> deact(source)

exclusion

empty boxes both can not be active at same time act(source) -> deact(target) act(target) -> deact(source)

arrow

STgroup · 2021-01-09 · Generated by Natural Docs

The End

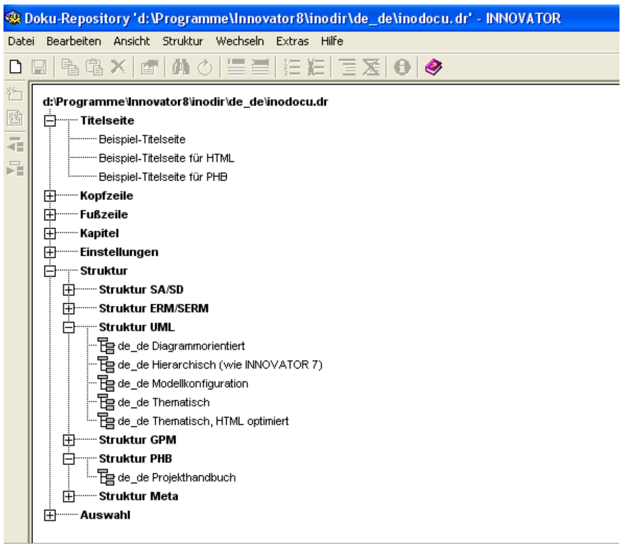
- Why is generation of documentation similar to code generation?
- Explain why a higher-order RAG is useful for documentation generation
- Which role does a pattern-matching language such as Xcerpt play in documentation generation?
- Why is the generation of documentation part of a macromodel?
- Why is a documentation a *derived model?*
- What happens if text from the API documentation flows back into the code as comments?

A.1 Other Template Expanders for Documentation Generation

Documentation Tools of MID Innovator

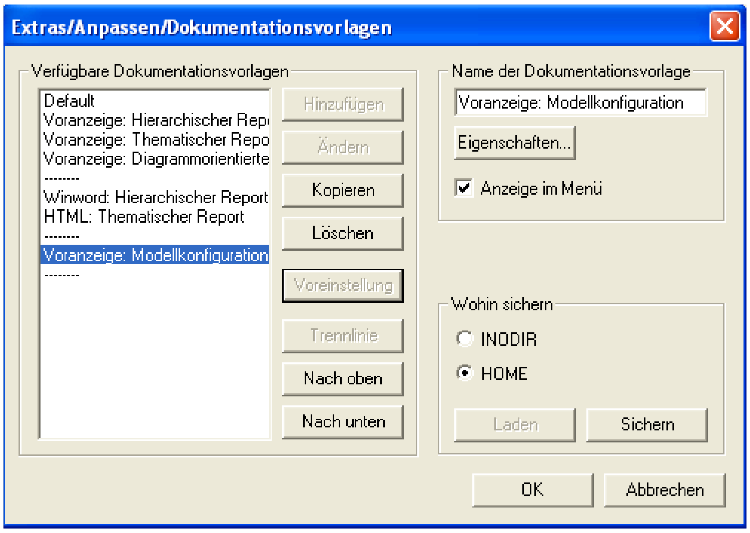
- Innovator provides documentation templates, into which diagrams, models, code can be embedded
- Several formats:
  - pdf
  - Word
  - ASCII
  - XML

Ex.: Innovator Documentation Template (Dokumentationsvorlage)


Doku-Repository 'd:\Programme\Innovator8\inodir\de_de\inodocu.dr' - INNOVATOR
Datei Bearbeiten Ansicht Struktur Wechseln Extras Hilfe

d:\Programme\Innovator8\inodir\de_de\inodocu.dr
- Titelseite
  - Beispiel-Titelseite
  - Beispiel-Titelseite für HTML
  - Beispiel-Titelseite für PHB
- Kopfzeile
- Fußzeile
- Kapitel
- Einstellungen
- Struktur
  - Struktur SA/SD
  - Struktur ERM/SERM
  - Struktur UML
    - de_de Diagrammorientiert
    - de_de Hierarchisch (wie INNOVATOR 7)
    - de_de Modellkonfiguration
    - de_de Thematisch
    - de_de Thematisch, HTML optimiert
  - Struktur GPM
  - Struktur PHB
    - de_de Projekthandbuch
  - Struktur Meta
- Auswahl

Ex.: Innovator Documentation Template (Dokumentationsvorlage): Adaptation


Extras/Anpassen/Dokumentationsvorlagen

Verfügbare Dokumentationsvorlagen

Default
Voranzeige: Hierarchischer Rep
Voranzeige: Thematischer Repo
Voranzeige: Diagrammorientierte

Winword: Hierarchischer Report
HTML: Thematischer Report

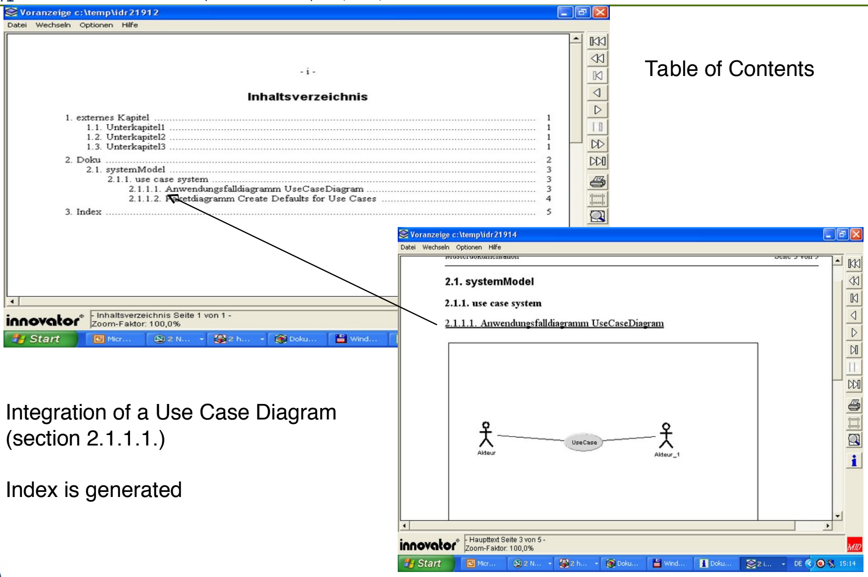
Voranzeige: Modellkonfiguration

Name der Dokumentationsvorlage: Voranzeige: Modellkonfiguration

[✓] Anzeige im Menü

Wohin sichern: INODIR [✓ HOME]

Innovator - Generated Example Word Document


Voranzeige c:\temp\idr21912
Datei Wechseln Optionen Hilfe

- i -

Inhaltsverzeichnis

1. externes Kapitel 1
   1.1. Unterkapitel1 1
   1.2. Unterkapitel2 1
   1.3. Unterkapitel3 1
2. Doku 2
   2.1. systemModel 3
      2.1.1. use case system 3
         2.1.1.1. Anwendungsfalldiagramm UseCaseDiagram 3
         2.1.1.2. Paketdiagramm Create Defaults for Use Cases 4
3. Index 5

- Inhaltsverzeichnis Seite 1 von 1 -
Zoom-Faktor: 100,0%

innovator
Start
Micr...
2 N...
2 h...
Doku...
Wind...
Voranzeige c:\temp\idr21914
Datei Wechseln Optionen Hilfe

2.1. systemModel

2.1.1. use case system

2.1.1.1. Anwendungsfalldiagramm UseCaseDiagram

- Haupttext Seite 3 von 5 -
Zoom-Faktor: 100,0%

Table of Contents

Integration of a Use Case Diagram (section 2.1.1.1.)

Index is generated