TeXstudio

Team 7 - Mark Boudreau, Faisal Rabbani, Sajad Wazin, Nicolas Courtemanche

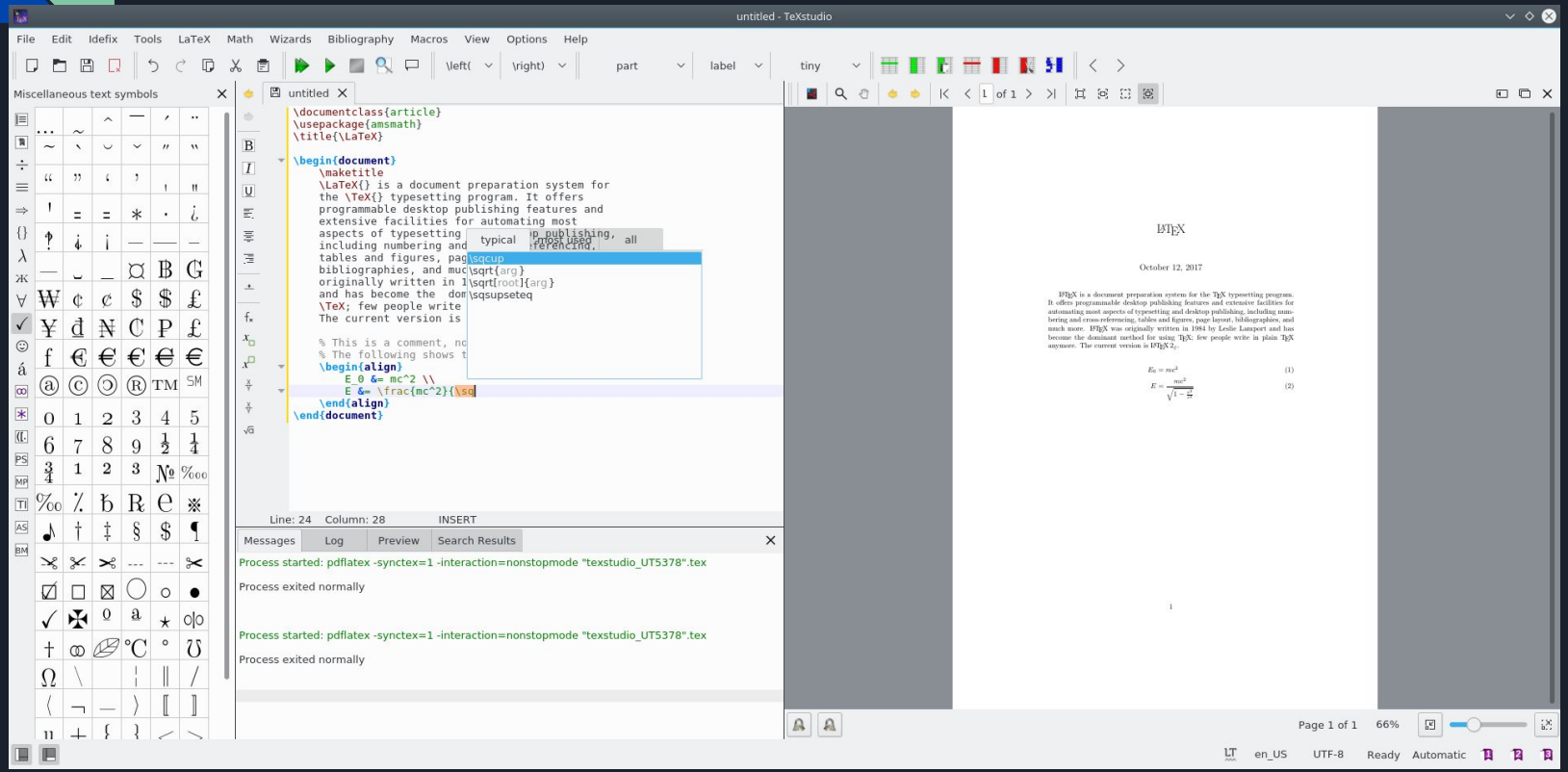
Presentation Objectives

- Introduction
- Functional View
- Information View
- Deployment View
- Apply Perspective
- Critiques

What is TeXstudio?

- Editor for (La)TeX documents
- Intends to make editing as easy as possible
- Provides bookmarking, autocompletion, assistants & wizards, mathematical symbol tables & lookup, table formatting, compilation output previews, and more
- Stakeholders: contributors to code and doc, users (Latex writers!)

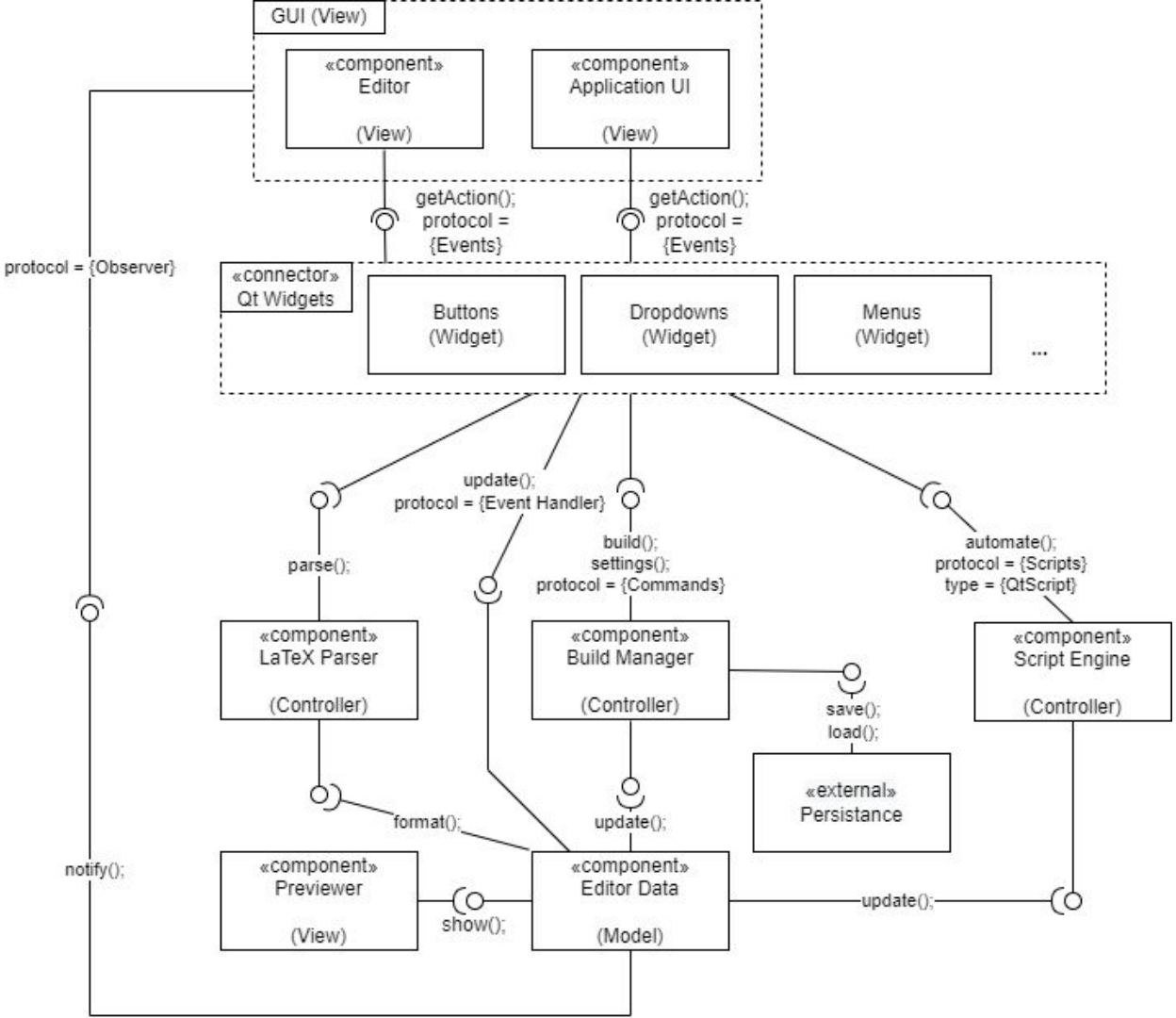
TeXstudio User Interface



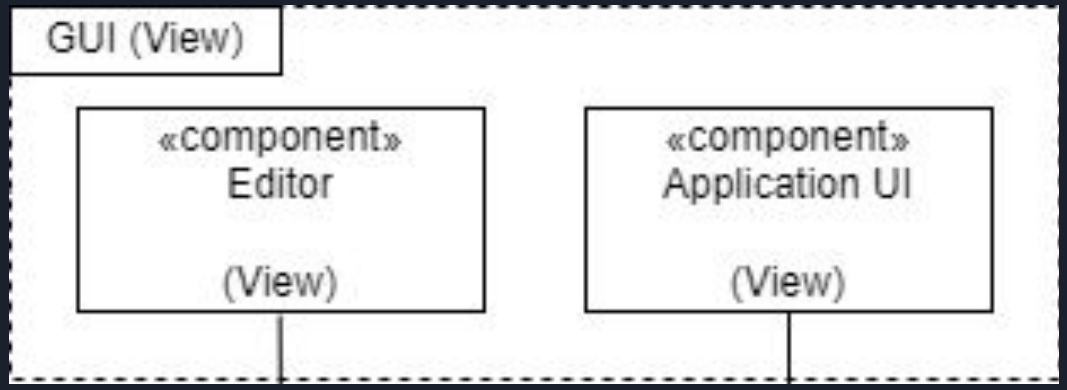
Source: https://en.wikipedia.org/wiki/TeXstudio#/media/File:TeXstudio_2.12.6.png (Alexander Wilms)

Functional View

- General Functional Structure
- Scope: Internal Interactions between elements
- Dependencies: Qt, Poppler and Phonon
- Flawed implementation of implicit M-V-C
- Heavily dependent on the Qt framework


GUI (View)
«component»
Editor
(View)
«component»
Application UI
(View)
getAction();
protocol =
{Events}
getAction();
protocol =
{Events}
protocol = {Observer}
«connector»
Qt Widgets
Buttons
(Widget)
Dropdowns
(Widget)
Menus
(Widget)
...
update();
protocol = {Event Handler}
parse();
build();
settings();
protocol = {Commands}
automate();
protocol = {Scripts}
type = {QtScript}
«component»
LaTeX Parser
(Controller)
«component»
Build Manager
(Controller)
«component»
Script Engine
(Controller)
save();
load();
«external»
Persistance
format();
update();
notify();
«component»
Previewer
(View)
«component»
Editor Data
(Model)
show();
update();

GUI (View)


GUI (View)
«component»
Editor
(View)
«component»
Application UI
(View)

C:\Users\Sajad\Desktop\McGill\McGill Winter 2022\Comp 252\Assignment 6\Q4.tex - TeXstudio

File Edit Idefix Tools LaTeX Math Wizards Bibliography Macros View Options Help

Q4.tex

```
    N[T_{i+1}] &= 2^{i+1}
\end{align*}
\hfill$\square$
\\
\noindent
Thus, to obtain a root-list with $T_0, T_1, ..., T_k$, we need to insert $\sum_{i=0}^{k} 2^k = 2^{k+1}-1$ nodes. However, we also need to account for the placeholder node $x$ that will be deleted, thus, in total we need to insert $2^{k+1}$ nodes.\\
\\
Using this method, you can obtain a Fibonacci Heap with $T_0, T_1, ..., T_k$ in the root-list, with $2^{k+1}-1$ nodes.
\clearpage
\noindent
(ii) We first start setting up this problem by creating a Fibonacci Heap. Let k be fixed and denote the size of elements in the root-list of our desired Fibonacci heap. Then the procedure for setting up the problem is as follows:
\begin{enumerate}
    \item Insert $2^{k+1}$ elements in the root-list
    \item Delete the last element that was added. Note that this will create a root-list with the structure $T_0, T_1, ..., T_k$.
    \item Loop through all nodes in the Fibonacci heap such that all nodes that aren't immediate children to the nodes in the root-list are deleted.
    \item Denote the remaining number of nodes by $n$.
\end{enumerate}
In fact, we can get an exact formula for the value of $n$:
\begin{align*}
    n &= k + \sum_{i=0}^{k} i\\
    n &= k + \dfrac{k^2 + k}{2}\\
    n &= \dfrac{3 \cdot k + k^2}{2}
\end{align*}
Note that deleting the number of non-immediate descendants in the Fibonacci Heap does \textit{not} affect the elements in the root-list. Thus, we also get the following inequation for $k$:
\begin{align*}
    3 \cdot k + k^2 &= 2n\\
    \sqrt{3 \cdot k + k^2} &= \sqrt{2n}\\
    \sqrt(k^2) &\leq \sqrt{2} * \sqrt{n}\\
    k &\leq \sqrt{2} * \sqrt{n}\\
\end{align*}
Since we have $k \leq c \cdot \sqrt{n}$ elements in the root-list, we can assume finding a minimum in the root-list has time complexity $\mathcal{O}(\sqrt{n})$. This completes the set-up for the problem. Then, we can repeat the following two steps to obtain the desired inequality:
\begin{enumerate}
    \item Insert a new minimum node $y$ with value $-\infty$
    \item Use DELETEMIN to delete $y$, which search for a new minimum node and also CLEAN-UP-ROOT-LIST.
\end{enumerate}
These two steps account for an operation each. In the first step, we simply insert a node into the root-list, thus this has $\mathcal{O}(1)$ time complexity. For the second operation, we need to take into account the cost of DELETEMIN, the cost of
```

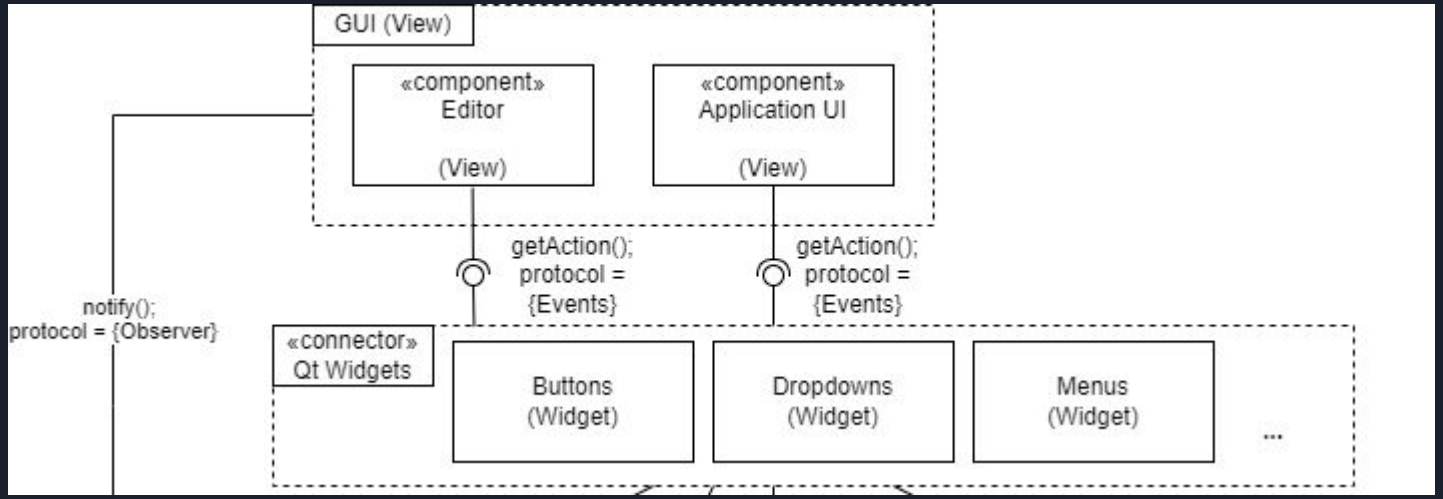
Line: 1 Column: 0 INSERT

Messages Log Preview Search Results

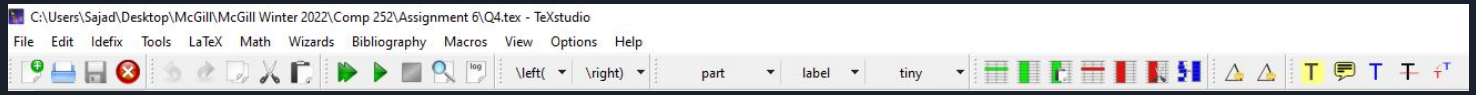
Editor Application UI

en_US UTF-8 Ready Automatic

GUI (View) and its interactions

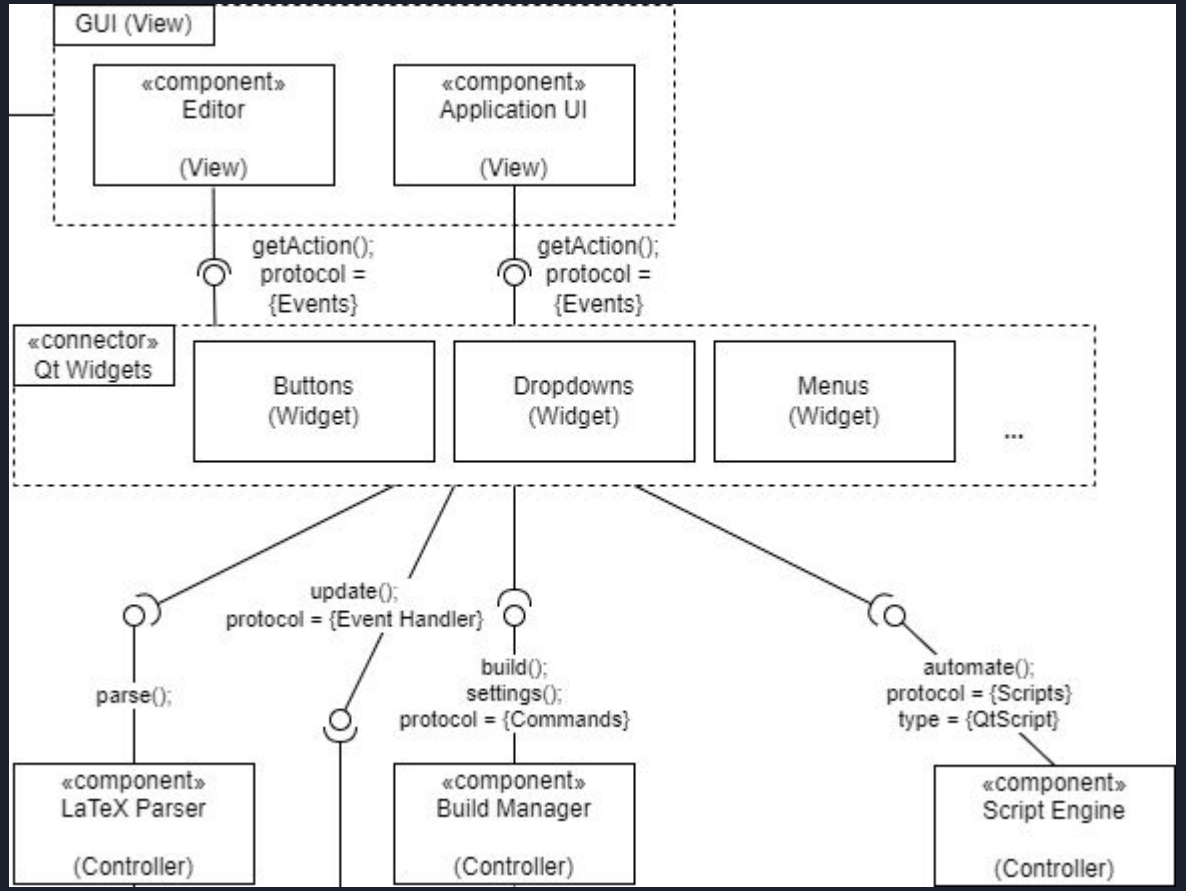


Qt Widgets



- Composite Design Pattern

Qt Widgets (Connector)



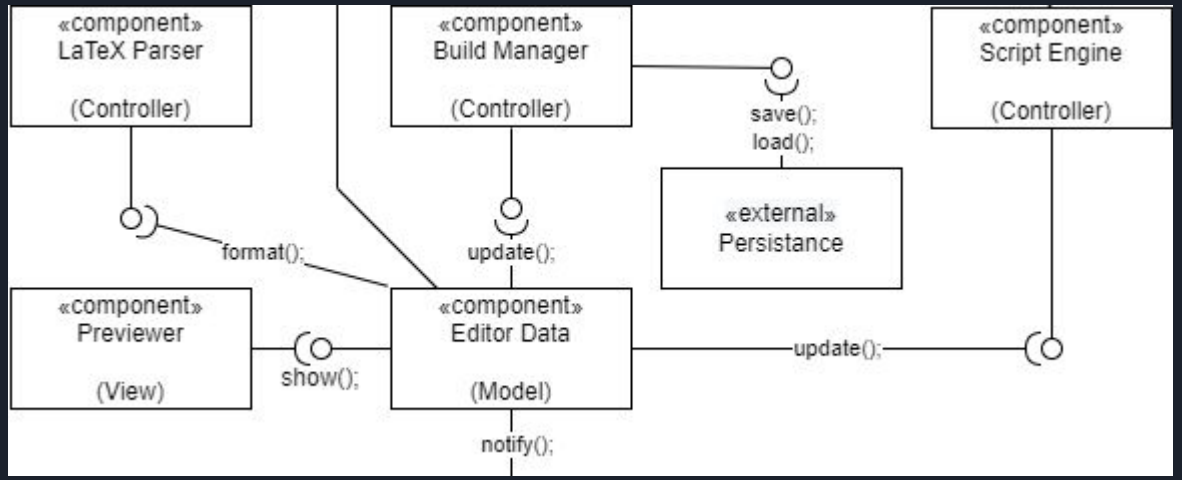
Important Notes about Qt Widgets

- Some Qt Widgets will interact directly with the Editor Data (Model)

- Some Qt Widgets use ***only*** functionality from Qt.

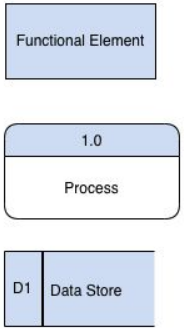
- There is a constructor method to easily create these widgets.

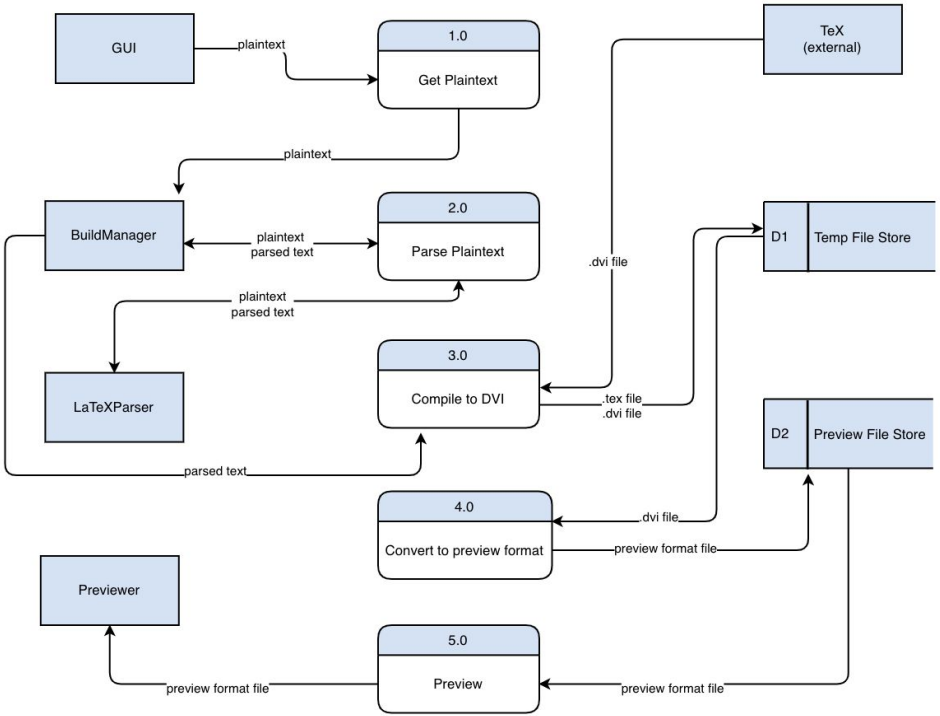
Event Processing Block



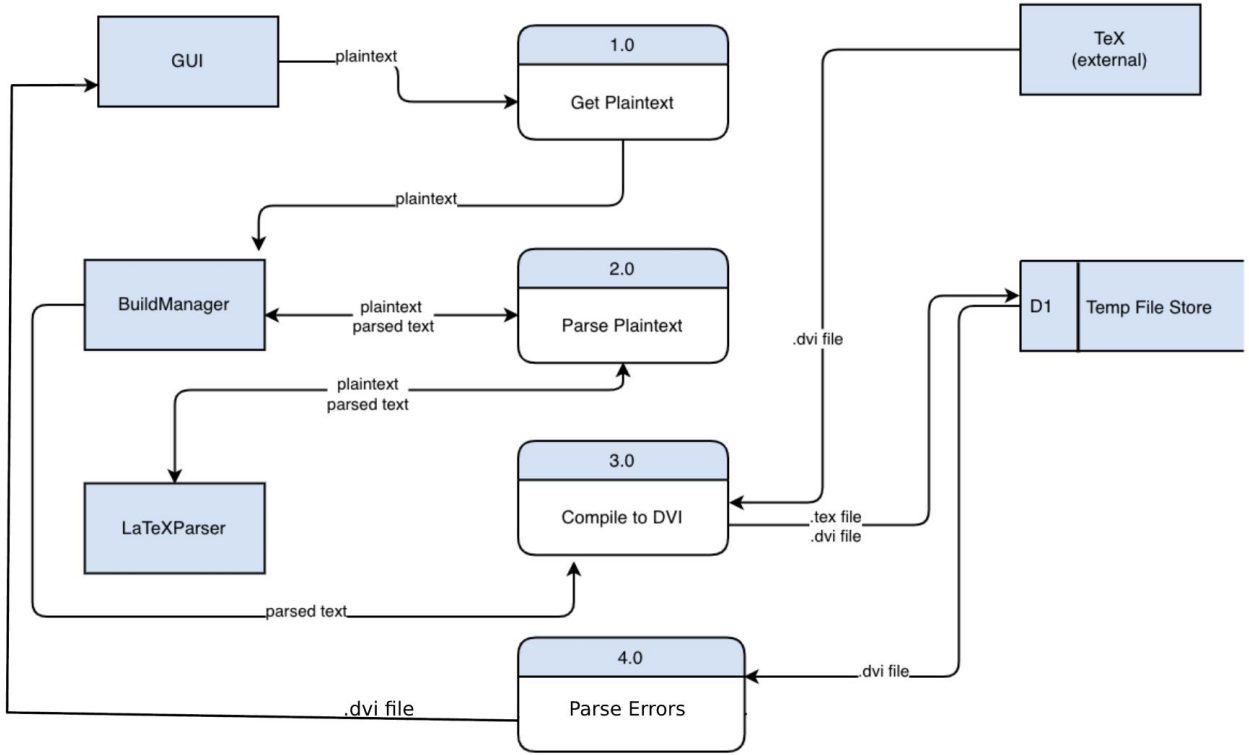
Information View

Legend





Failure Scenario



Deployment View

- Most deployments of TeXstudio are quite simple, just run the installer for your OS and enjoy!
- Some advanced features require extra setup. For example, making use of a LanguageTool server
- LanguageTool provides grammar & style checking as an HTTP service
- TeXstudio can use LanguageTool to provide grammar hints (squiggly blue underlines) in the editor buffer
- If run locally, TeXstudio can manage LanguageTool itself, just need a JRE installation (8 or later) and the LanguageTool JAR file.
- If run remote, need JRE on the host, must start the server manually. On the client, configure TeXstudio to use the remote host

Perspective: Usability

Three major concerns and tactics towards **Functional**, **Information** and **Deployment** views:

	Functional	**Information**	**Deployment**
Simplicity	UI simplicity	-	Easy modulation
Consistency	Platform agnosticity (internal)	CVS to work with GIT	Platform agnosticity (external)
Efficiency	Efficient UI	Efficient compilation, data flow for humans	Small deployment package

Critiques

Critique 1: God Class Antipattern + Single Responsibility Principle

- TexStudio.cpp: 12,000+ lines
  - Mapped in our AD to different functional elements:
    - Qt Widget
    - GUI
    - Persistence
    - Previewer
- BuildManager.cpp: 2,500+ lines
  - Broad scope
  - Responsibilities: management of document lifecycle
  - Also includes implemented concurrency functions:
    - BuildManager::firstProcessOfDirectExpansion
- /src contains **97 .cpp files!**

Critiques

Critique 2: Inappropriate Intimacy

- Many cases of methods using public fields in other classes
- This is a failure to encapsulate: Class implementation not hidden
- Texstudio class: At least 192 references to public fields in ConfigManager class. Likely to have more examples of this
- Refactoring ConfigManager may break the interface expected by Texstudio, thus requiring even more refactoring
- Should use getter & setter methods, hides the underlying implementation, makes refactoring easier

Questions?