A Web-Based Toolkit for Mathematical Word Processing Applications with Semantics

Nathan Carter* and Kenneth G. Monks

Joint Mathematics Meetings, January 10, 2018

Talk Outline

1. Background on *Lurch* project
2. Description of the *Lurch Web Platform*
3. Demo applications

Goal of *Lurch*

An educational proof checker for student use in a first proof course

- No automated theorem proving—the student should do the work
- No need for a powerful engine—e.g., dependent type theory
- No code—user interface must be just like books, notes, PDFs

Current release: Version 0.8 for Windows, Mac, Linux

Consequences of the Goal

- Must be a plain word processor (not a code IDE)
- Must use typeset mathematics, like math texts do
- Must permit the user to define mathematical systems
- Must permit the user to define mathematical notation

Consequences of the Goal

- Must be a plain word processor (not a code IDE)
  Implemented in *Lurch* 0.8 ✓
- Must use typeset mathematics, like math texts do
- Must permit the user to define mathematical systems
- Must permit the user to define mathematical notation

Consequences of the Goal

- Must be a plain word processor (not a code IDE)
  Implemented in *Lurch* 0.8 ✓
- Must use typeset mathematics, like math texts do
  Typeset output in *Lurch* 0.8 ✓
- Must permit the user to define mathematical systems
- Must permit the user to define mathematical notation

Consequences of the Goal

- Must be a plain word processor (not a code IDE)
  Implemented in *Lurch* 0.8 ✓
- Must use typeset mathematics, like math texts do
  Typeset output in *Lurch* 0.8 ✓
- Must permit the user to define mathematical systems
  Implemented in *Lurch* 0.8 ✓
- Must permit the user to define mathematical notation

Consequences of the Goal

- Must be a plain word processor (not a code IDE)
  Implemented in *Lurch* 0.8 ✓
- Must use typeset mathematics, like math texts do
  Typeset output in *Lurch* 0.8 ✓
- Must permit the user to define mathematical systems
  Implemented in *Lurch* 0.8 ✓
- Must permit the user to define mathematical notation
  Not implemented in *Lurch* 0.8 ×

Theorem: *The composition of injections is an injection.*

Proof:

Assume $f : A \to B$, f is $injective$, $g : B \to C$, and g is $injective$. Then we know that $g \circ f : A \to C$ and $\forall x, g \circ f(x) = g(f(x))$, because that's what function composition means when $f : A \to B$ and $g : B \to C$.

So let x and y be arbitrary elements of A. From $\forall x, g \circ f(x) = g(f(x))$, we can obtain both $g \circ f(x) = g(f(x))$ and $g \circ f(y) = f(f(y))$, by the definition of the quantifier $\forall$.

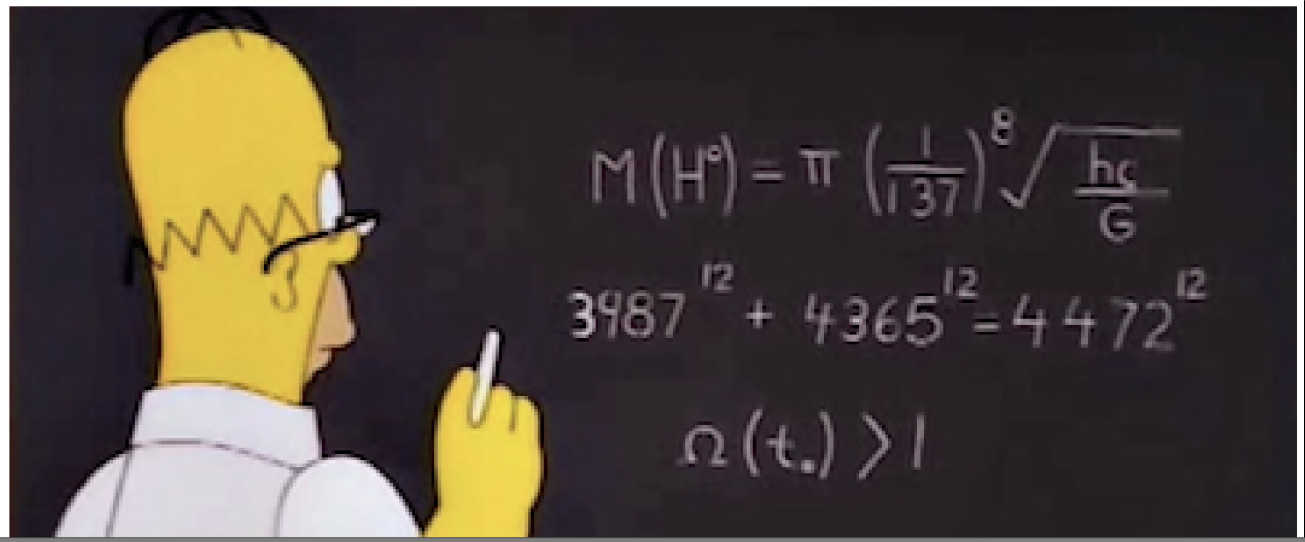
wp-screenshots – Lurch

183% DejaVu Sans 12.00 none Find:

Lurch is a mathematical word processor.

It has support for math typesetting, such as $\forall x, x \neq 0 \Rightarrow \left(\exists y, \frac{x}{y} = 1\right)$.

It also supports *various fonts*, styles, and images:



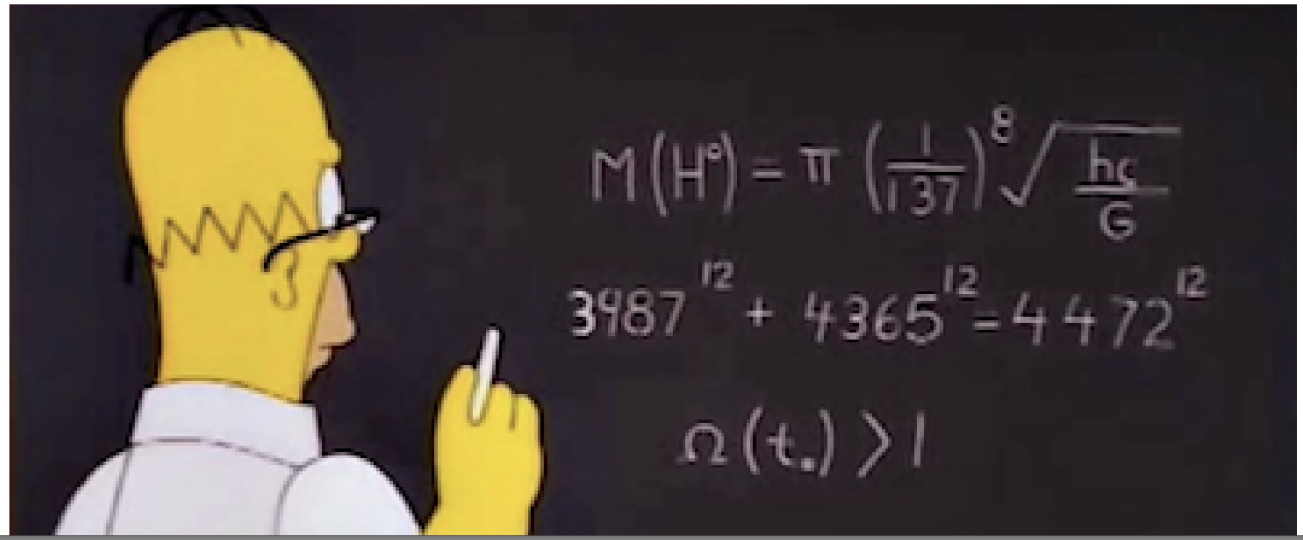
wp-screenshots – Lurch

183% DejaVu Sans 12.00 none Find:

Lurch is a mathematical word processor.

It has support for math typesetting, such as $\forall x, x \neq 0 \Rightarrow \left(\exists y, \frac{x}{y} = 1\right)$.

It also supports *various fonts*, styles, and images:



wp-screenshots – Lurch

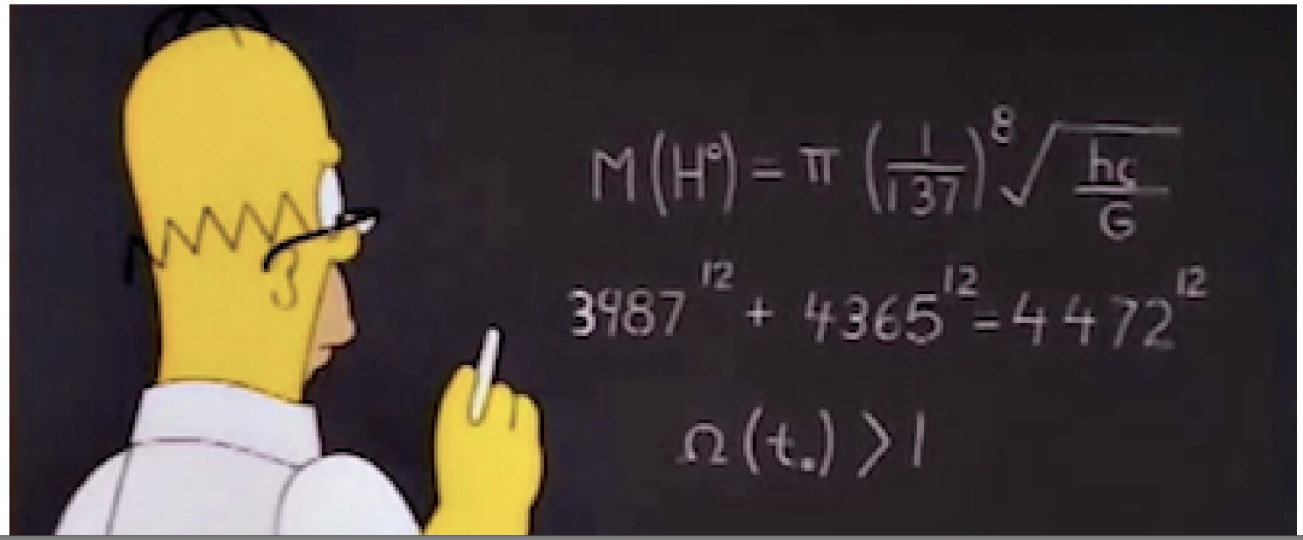
183% DejaVu Sans 12.00 none Find:

Lurch is a mathematical word processor.

∀ *quantification*

It has support for math typesetting, such as $\forall x, x \neq 0 \Rightarrow \left(\exists y, \frac{x}{y} = 1\right)$.

It also supports *various fonts*, styles, and images:



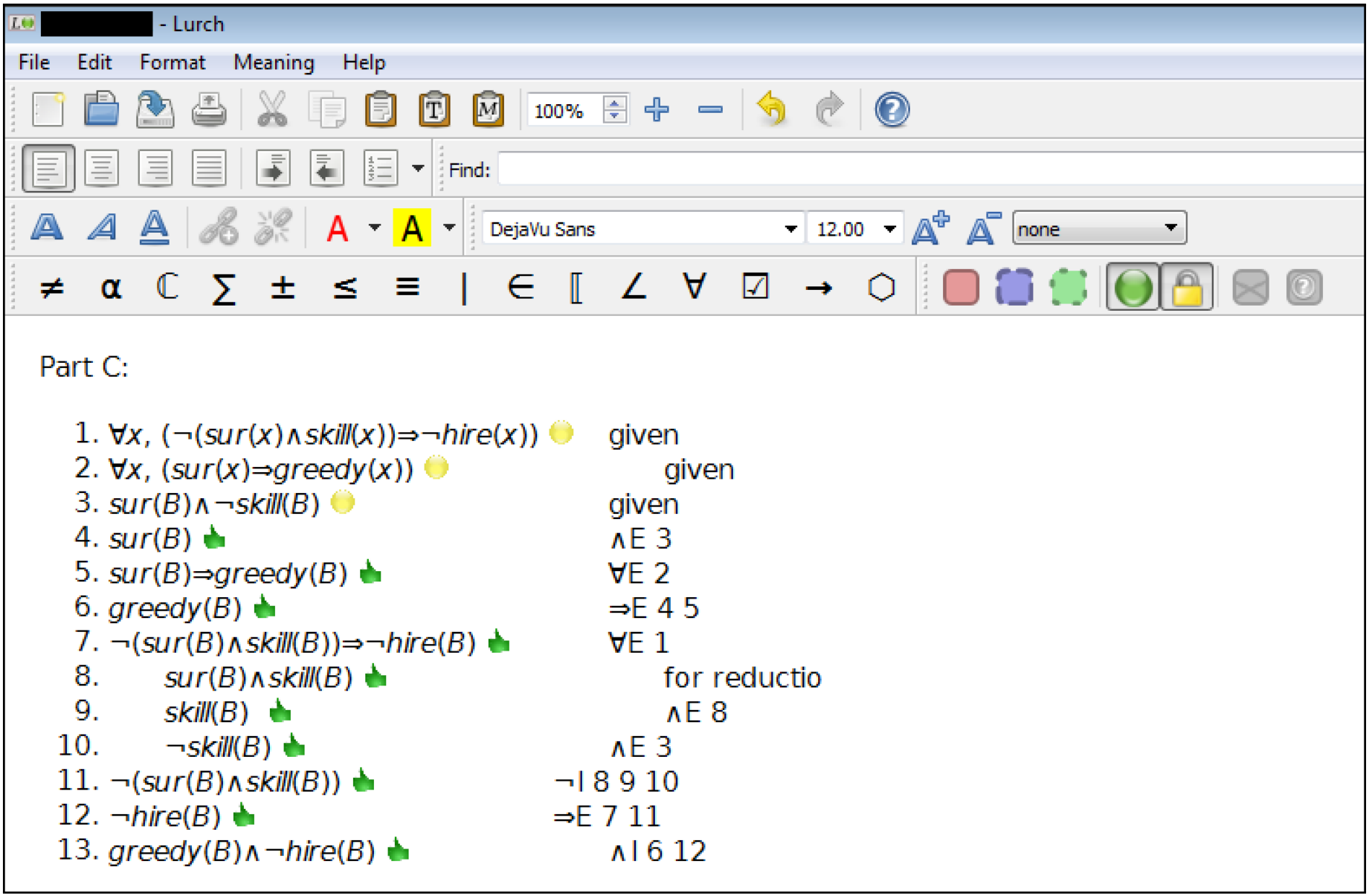
Lyx:

$$+\alpha\left[R_{ij}-2K_{il}K^{l}_{j}+KK_{ij}-S_{ij}-\frac{1}{2}\gamma_{ij}(\rho-S)\right]. \qquad (3)\ \boxed{\text{K dot}}$$

$$\Gamma^{i}_{jk}=\frac{\gamma^{il}}{2}(\gamma_{lj,k}+\gamma_{lk,j}+\gamma_{jk,l}),$$

Word:

$$\int_{a}^{b} f(t)dt$$



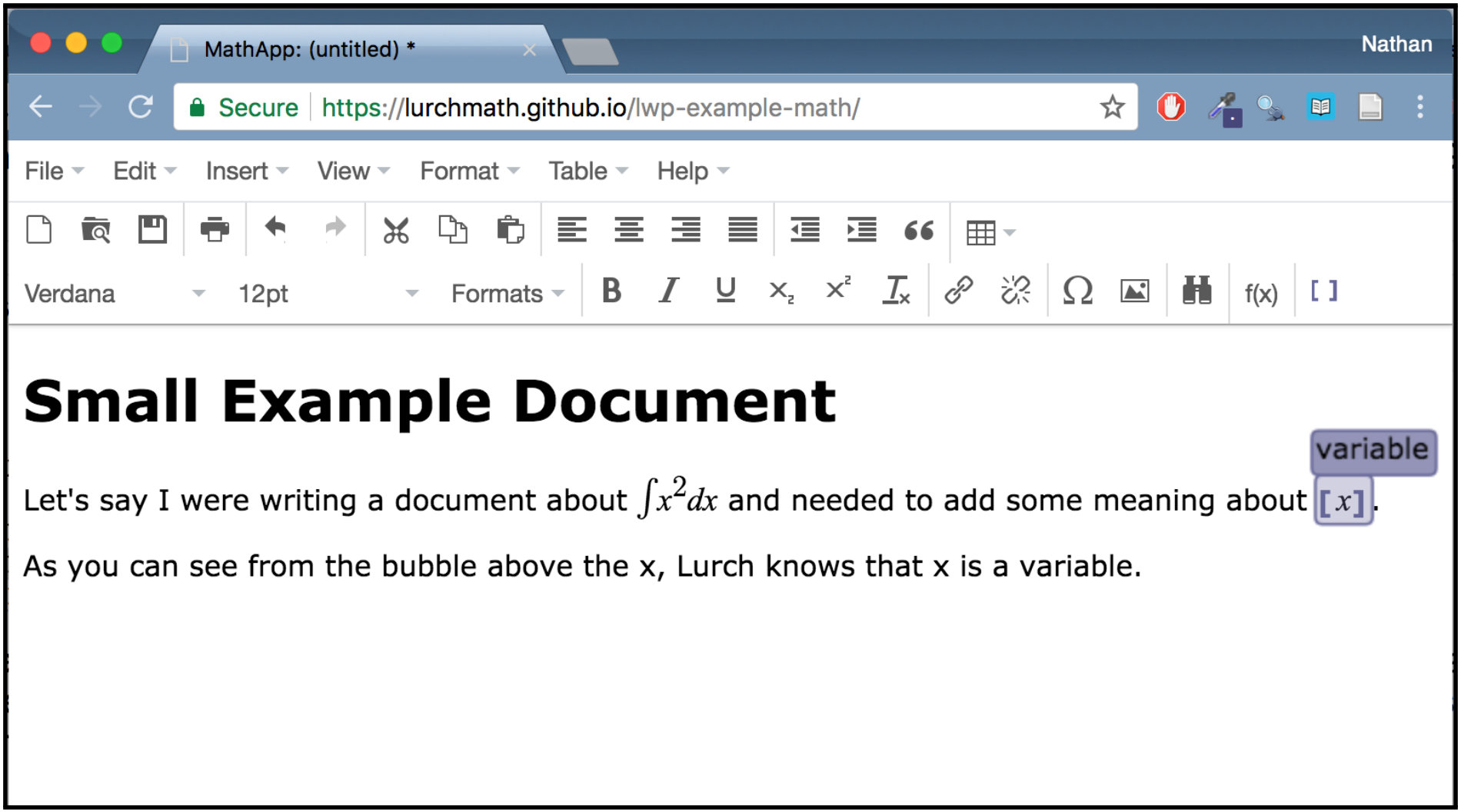
Part C:

1.	$\forall x, (\neg(sur(x) \wedge skill(x)) \Rightarrow \neg hire(x))$	given	
2.	$\forall x, (sur(x) \Rightarrow greedy(x))$	given	
3.	$sur(B) \wedge \neg skill(B)$	given	
4.	$sur(B)$	$\wedge$E 3	
5.	$sur(B) \Rightarrow greedy(B)$	$\forall$E 2	
6.	$greedy(B)$	$\Rightarrow$E 4 5	
7.	$\neg(sur(B) \wedge skill(B)) \Rightarrow \neg hire(B)$	$\forall$E 1	
8.	$sur(B) \wedge skill(B)$	for reductio	
9.	$skill(B)$	$\wedge$E 8	
10.	$\neg skill(B)$	$\wedge$E 3	
11.	$\neg(sur(B) \wedge skill(B))$	$\neg$I 8 9 10	
12.	$\neg hire(B)$	$\Rightarrow$E 7 11	
13.	$greedy(B) \wedge \neg hire(B)$	$\wedge$I 6 12	

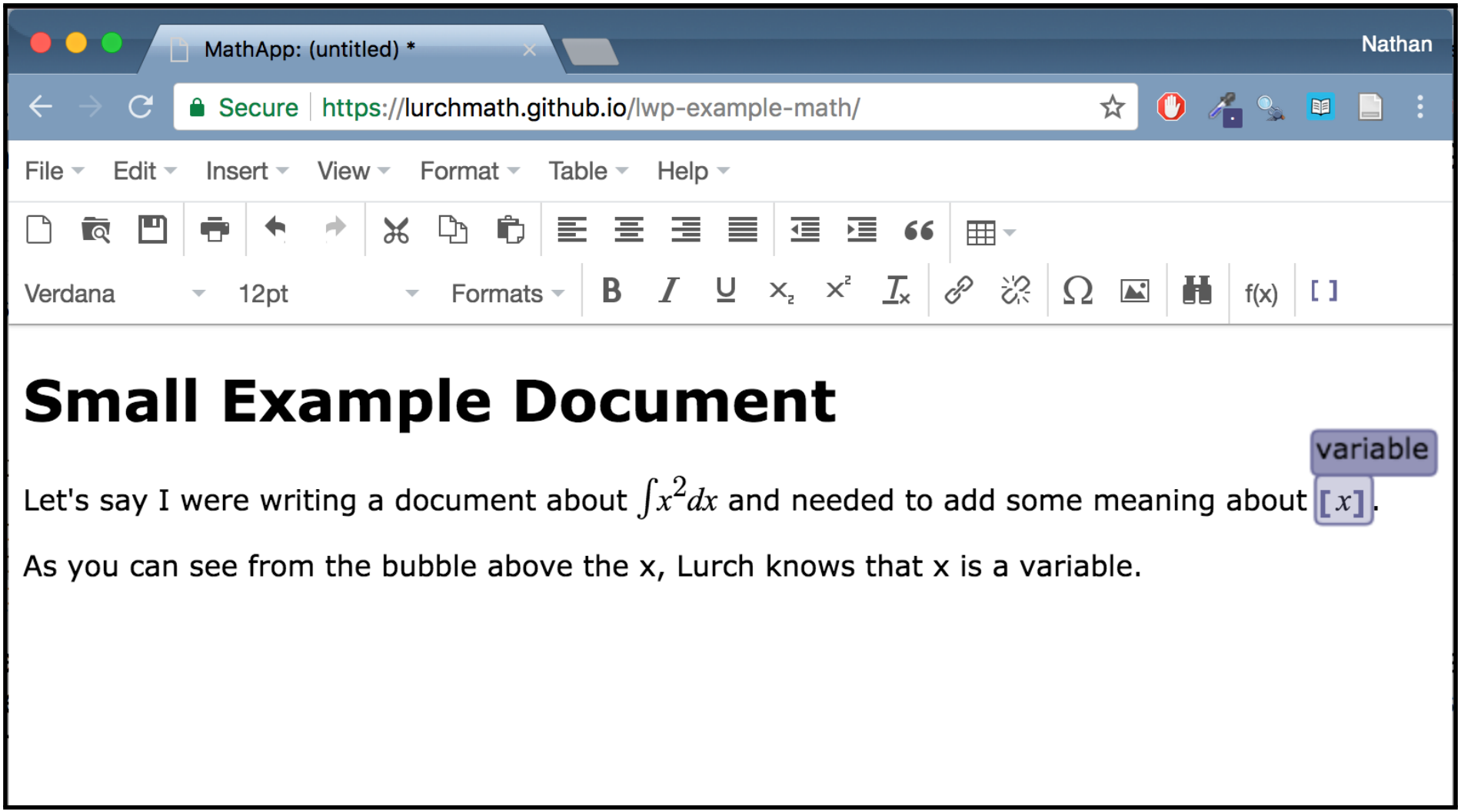
Limitations of version 0.8

- **Interface:** Feedback from users has generated many new user interface ideas and principles that will improve the UX.
- **Engine:** *Lurch*'s reasoning system can benefit from a redesign incorporating what we've learned from classroom use and from the ATP research community.
- **Platform:** Since we began work on the desktop version (2008) browsers have become more powerful. There are many advantages to moving to the web, including accessibility.
- **Inefficiency:** Desktop *Lurch* is slow, due to a single-threaded technology on which much of the desktop version was built.

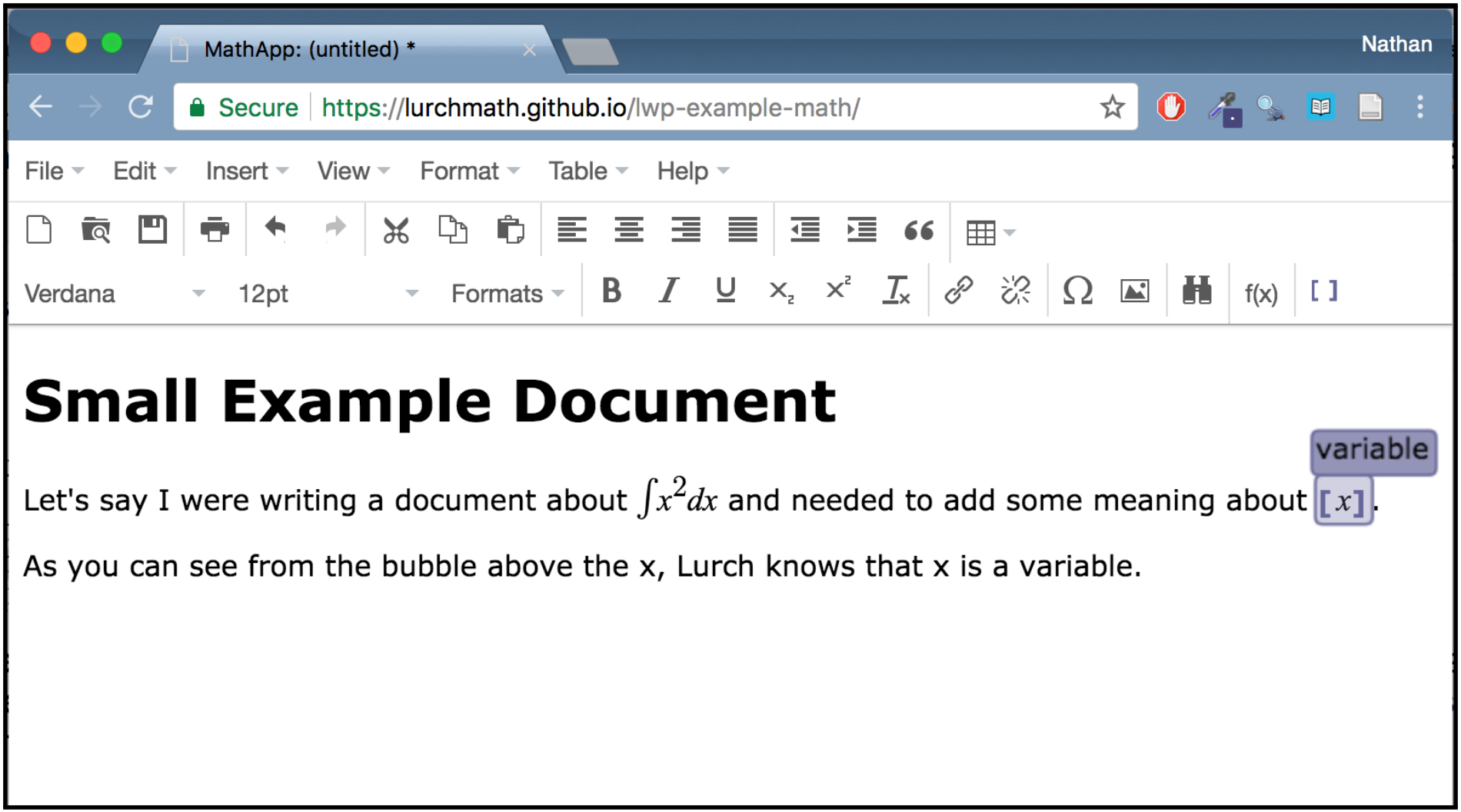
The *Lurch Web Platform* is a set of JavaScript code for building web applications built on a mathematical word processor with semantics.



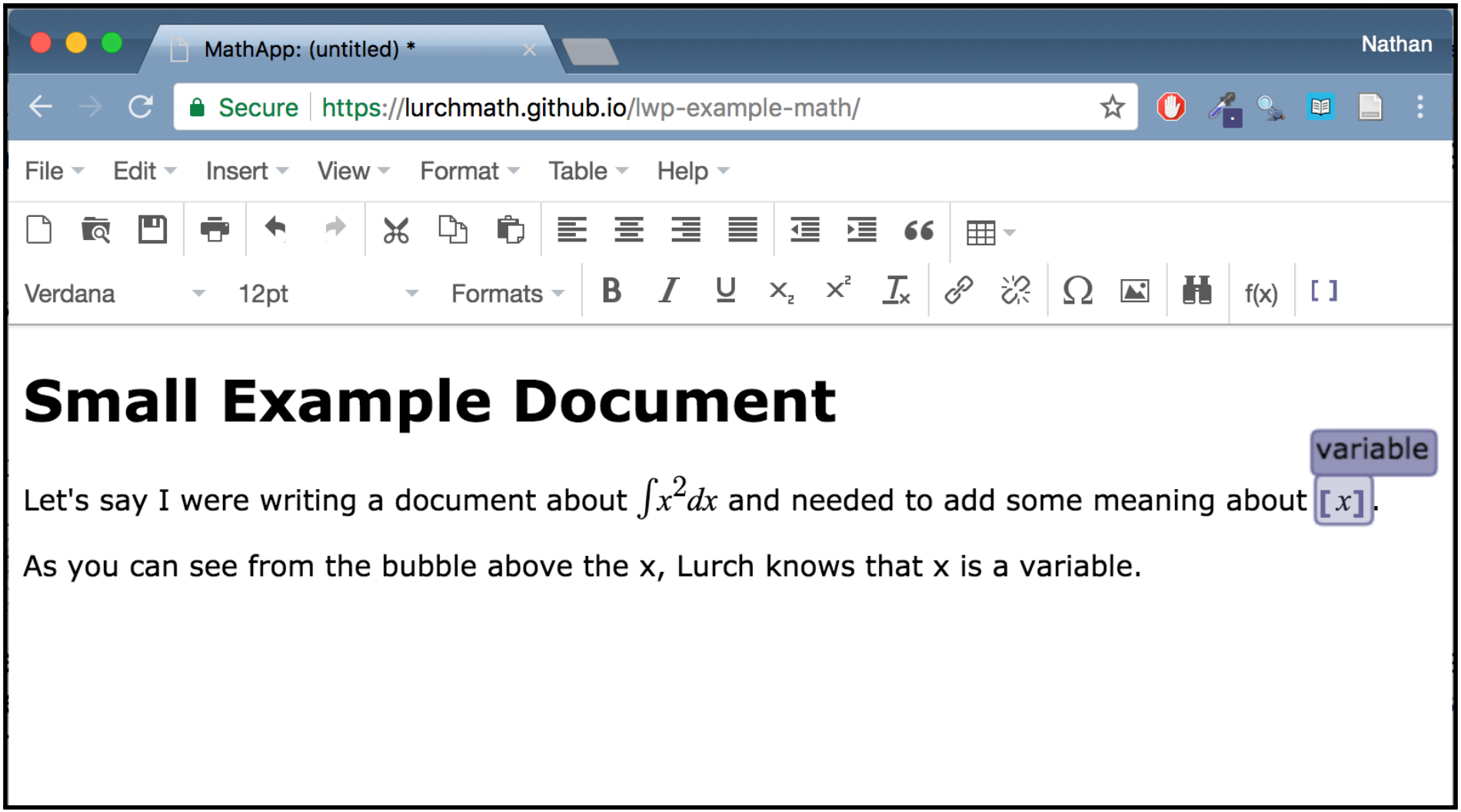
The *Lurch Web Platform* is a set of JavaScript code for building web applications built on a mathematical word processor with semantics.



The *Lurch Web Platform* is a set of JavaScript code for building web applications built on a mathematical word processor with semantics.



The *Lurch Web Platform* is a set of JavaScript code for building web applications built on a mathematical word processor with semantics.



Building a web version: Two Phases

1. **User Interface:**
   This is the *Lurch Web Platform*.
   It is re-usable in other applications.

2. **Reasoning system:**
   Goal: Finish design in AY17–18 (Monks sabbatical),
   implement in AY18–19 (Carter sabbatical).

Building a web version: Two Phases

1. **User Interface:** Complete ✓
   This is the *Lurch Web Platform*.
   It is re-usable in other applications.

2. **Reasoning system:**
   Goal: Finish design in AY17–18 (Monks sabbatical),
   implement in AY18–19 (Carter sabbatical).

Building a web version: Two Phases

1. **User Interface:** Complete ✓
   This is the *Lurch Web Platform*.
   It is re-usable in other applications.

2. **Reasoning system:** Ongoing design and implementation
   Goal: Finish design in AY17–18 (Monks sabbatical),
   implement in AY18–19 (Carter sabbatical).

Features

- Usual features of a word processor (built on TinyMCE)
- Typeset mathematics (form, not content, built on MathQuill)
- Bubbles (arbitrarily many, nested, various types)
- Connections among bubbles
- OpenMath support
- Wiki integration for cloud storage and sharing of documents
- Importing meaning from other documents (“dependencies”)
- Dropbox integration
- Background threads for long computations
- etc.

Features

- Usual features of a word processor (built on TinyMCE)
- Typeset mathematics (form, not content, built on MathQuill)
- Bubbles (arbitrarily many, nested, various types)
- Connections among bubbles
- OpenMath support
- Wiki integration for cloud storage and sharing of documents
- Importing meaning from other documents (“dependencies”)
- Dropbox integration
- Background threads for long computations
- etc.

Features

- Usual features of a word processor (built on TinyMCE)
- Typeset mathematics (form, not content, built on MathQuill)
- Bubbles (arbitrarily many, nested, various types)
- Connections among bubbles
- OpenMath support
- Wiki integration for cloud storage and sharing of documents
- Importing meaning from other documents (“dependencies”)
- Dropbox integration
- Background threads for long computations
- etc.

Features

- Usual features of a word processor (built on TinyMCE)
- Typeset mathematics (form, not content, built on MathQuill)
- Bubbles (arbitrarily many, nested, various types)
- Connections among bubbles
- OpenMath support
- Wiki integration for cloud storage and sharing of documents
- Importing meaning from other documents (“dependencies”)
- Dropbox integration
- Background threads for long computations
- etc.

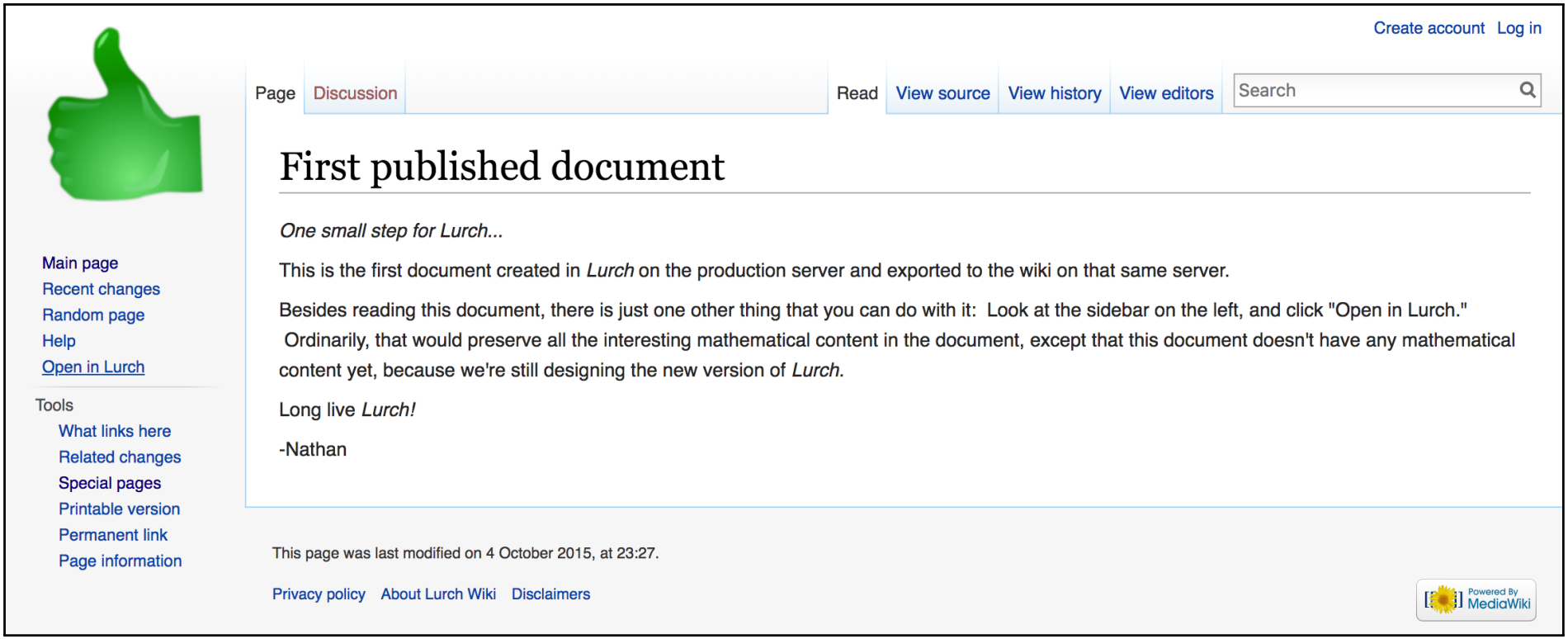
Features

- Usual features of a word processor (built on TinyMCE)
- Typeset mathematics (form, not content, built on MathQuill)
- Bubbles (arbitrarily many, nested, various types)
- Connections among bubbles
- OpenMath support
- Wiki integration for cloud storage and sharing of documents
- Importing meaning from other documents (“dependencies”)
- Dropbox integration
- Background threads for long computations
- etc.

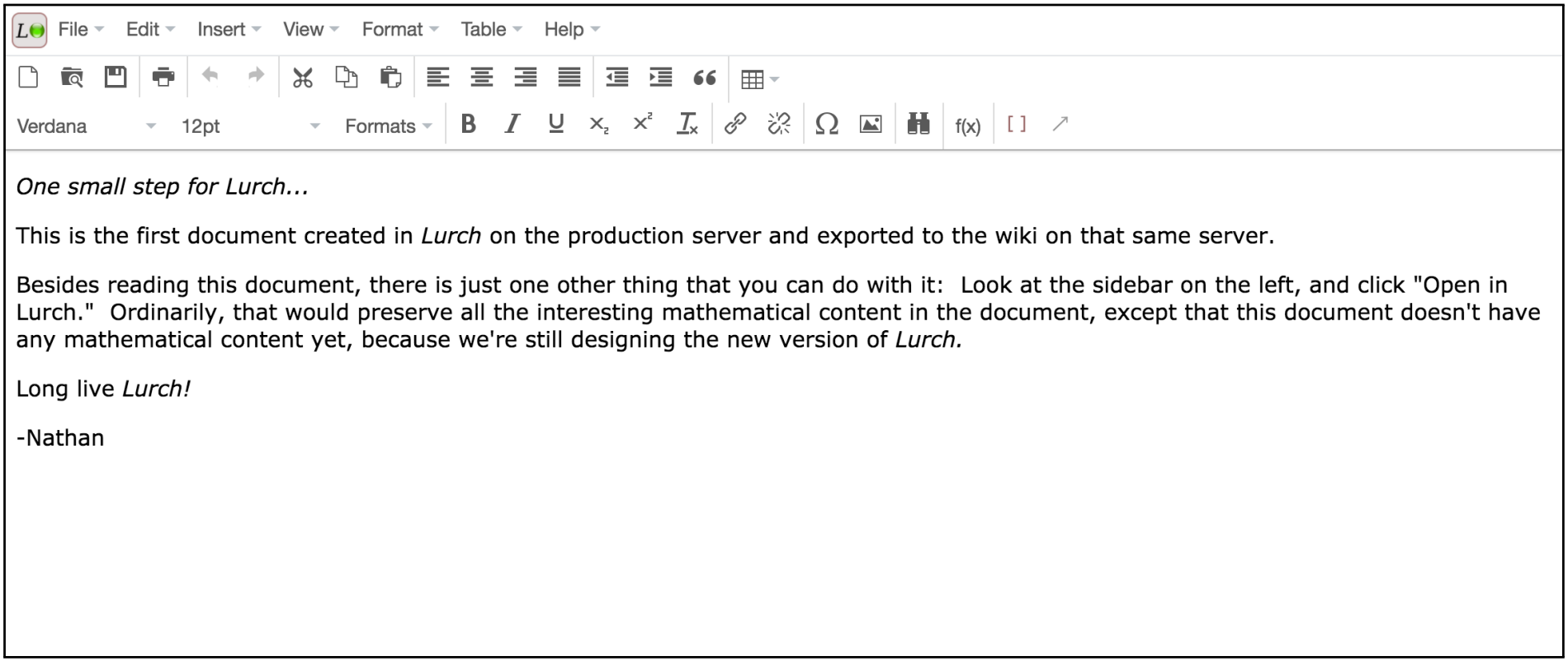
Features

- Usual features of a word processor (built on TinyMCE)
- Typeset mathematics (form, not content, built on MathQuill)
- Bubbles (arbitrarily many, nested, various types)
- Connections among bubbles
- OpenMath support
- Wiki integration for cloud storage and sharing of documents
- Importing meaning from other documents (“dependencies”)
- Dropbox integration
- Background threads for long computations
- etc.

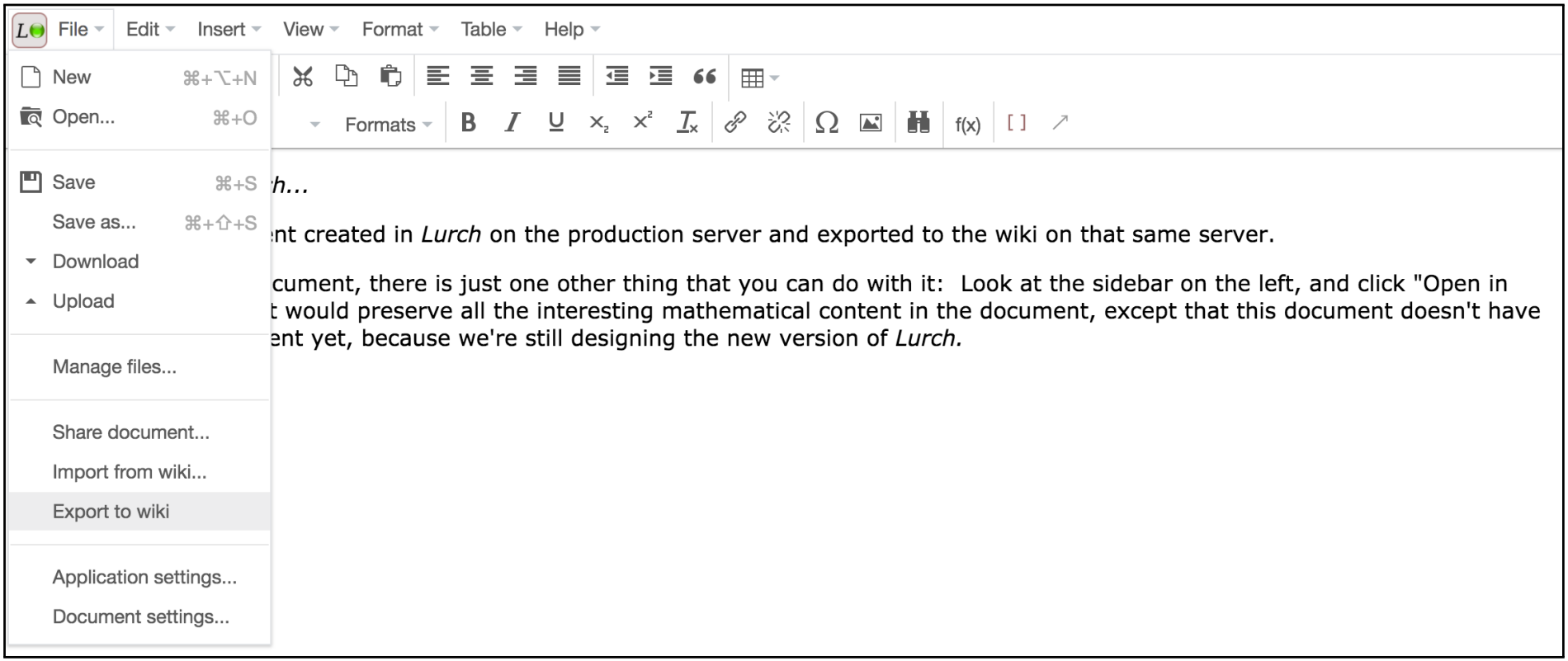
Features


Create account
Log in
Page
Discussion
Read
View source
View history
View editors
Search
First published document
One small step for Lurch...
This is the first document created in Lurch on the production server and exported to the wiki on that same server.
Besides reading this document, there is just one other thing that you can do with it: Look at the sidebar on the left, and click "Open in Lurch." Ordinarily, that would preserve all the interesting mathematical content in the document, except that this document doesn't have any mathematical content yet, because we're still designing the new version of Lurch.
Long live Lurch!
-Nathan
Main page
Recent changes
Random page
Help
Open in Lurch
Tools
What links here
Related changes
Special pages
Printable version
Permanent link
Page information
This page was last modified on 4 October 2015, at 23:27.
Privacy policy
About Lurch Wiki
Disclaimers
Powered By MediaWiki

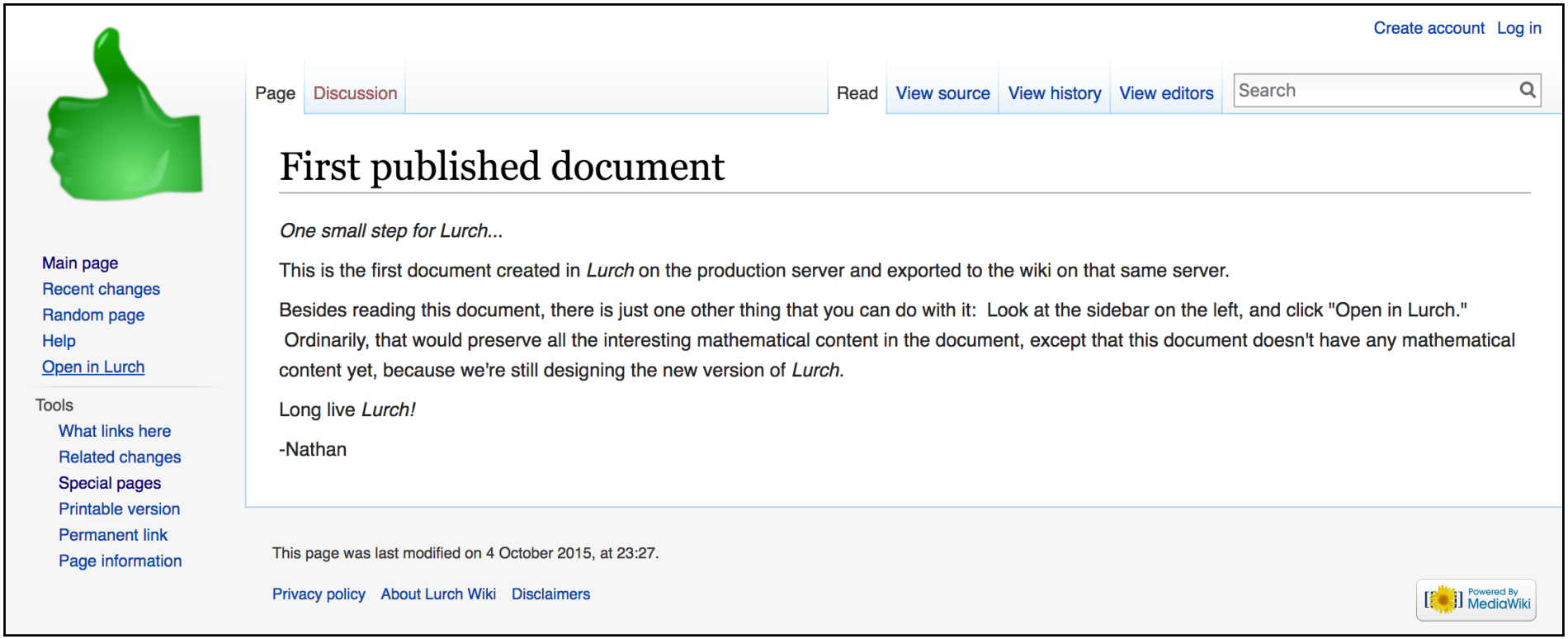
Features


File
Edit
Insert
View
Format
Table
Help
Verdana
12pt
Formats
f(x)
One small step for Lurch...
This is the first document created in Lurch on the production server and exported to the wiki on that same server.
Besides reading this document, there is just one other thing that you can do with it: Look at the sidebar on the left, and click "Open in Lurch." Ordinarily, that would preserve all the interesting mathematical content in the document, except that this document doesn't have any mathematical content yet, because we're still designing the new version of Lurch.
Long live Lurch!
-Nathan

Features


File
Edit
Insert
View
Format
Table
Help
New ⌘+⌥+N
Open... ⌘+O
Save ⌘+S
Save as... ⌘+⇧+S
Download
Upload
Manage files...
Share document...
Import from wiki...
Export to wiki
Application settings...
Document settings...
Formats
f(x)
nt created in Lurch on the production server and exported to the wiki on that same server.
cument, there is just one other thing that you can do with it: Look at the sidebar on the left, and click "Open in
t would preserve all the interesting mathematical content in the document, except that this document doesn't have
ent yet, because we're still designing the new version of Lurch.

Features


Create account
Log in
Page
Discussion
Read
View source
View history
View editors
Search
First published document
One small step for Lurch...
This is the first document created in Lurch on the production server and exported to the wiki on that same server.
Besides reading this document, there is just one other thing that you can do with it: Look at the sidebar on the left, and click "Open in Lurch." Ordinarily, that would preserve all the interesting mathematical content in the document, except that this document doesn't have any mathematical content yet, because we're still designing the new version of Lurch.
Long live Lurch!
-Nathan
Main page
Recent changes
Random page
Help
Open in Lurch
Tools
What links here
Related changes
Special pages
Printable version
Permanent link
Page information
This page was last modified on 4 October 2015, at 23:27.
Privacy policy
About Lurch Wiki
Disclaimers
Powered By MediaWiki

Features

- Usual features of a word processor (built on TinyMCE)
- Typeset mathematics (form, not content, built on MathQuill)
- Bubbles (arbitrarily many, nested, various types)
- Connections among bubbles
- OpenMath support
- Wiki integration for cloud storage and sharing of documents
- Importing meaning from other documents (“dependencies”)
- Dropbox integration
- Background threads for long computations
- etc.

Features

- Usual features of a word processor (built on TinyMCE)
- Typeset mathematics (form, not content, built on MathQuill)
- Bubbles (arbitrarily many, nested, various types)
- Connections among bubbles
- OpenMath support
- Wiki integration for cloud storage and sharing of documents
- Importing meaning from other documents (“dependencies”)
- Dropbox integration
- Background threads for long computations
- etc.

Features

- Usual features of a word processor (built on TinyMCE)
- Typeset mathematics (form, not content, built on MathQuill)
- Bubbles (arbitrarily many, nested, various types)
- Connections among bubbles
- OpenMath support
- Wiki integration for cloud storage and sharing of documents
- Importing meaning from other documents (“dependencies”)
- Dropbox integration
- Background threads for long computations
- etc.

Demo Applications

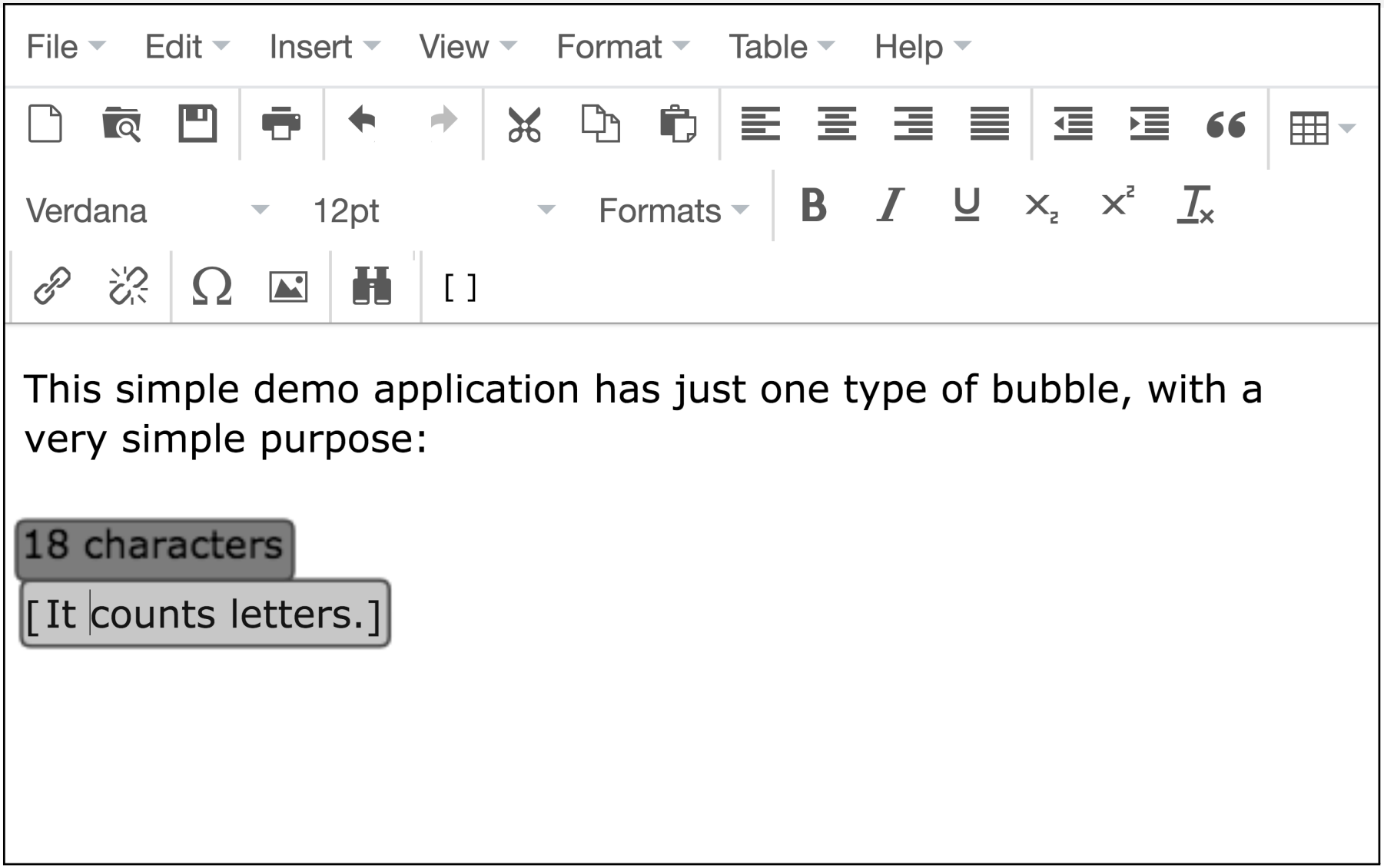
Two purposes:

1. Give this audience an idea of what can be done with the toolkit
2. Give developers example code to imitate/reuse in their own work

Applications:

- Simple Example — “Hello World”
- Complex Example — one step above Simple Example
- Math Evaluator — numerical evaluation of simple formulas
- OMCD Editor — creates a specific type of XML structure
- Code Editor — creates programs in a tiny language
- Lean UI — creates programs/proofs checked by Lean
- Main *Lurch* app — just beginning

Demo Application: Simple Example



Demo Applications

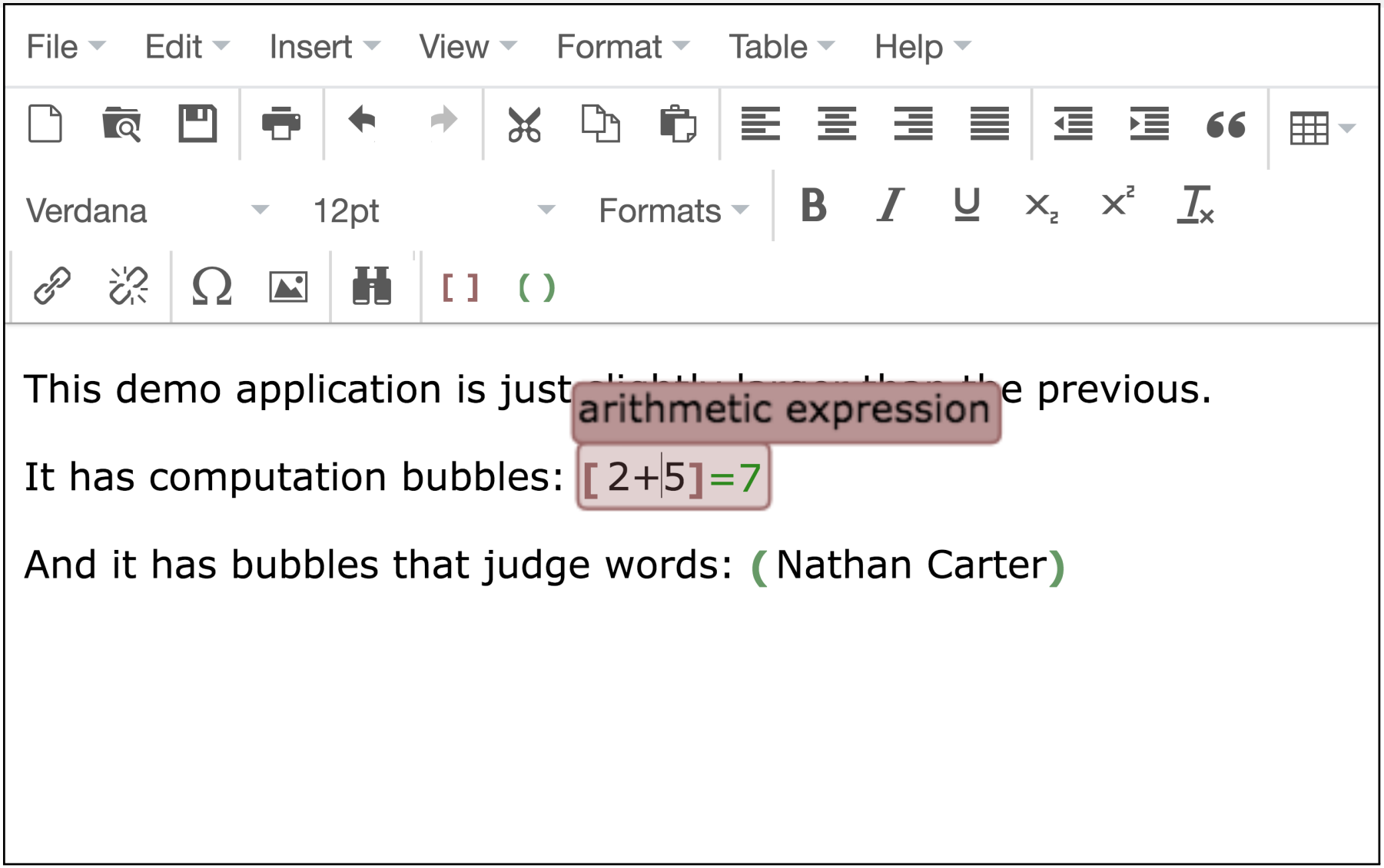
Two purposes:

1. Give this audience an idea of what can be done with the toolkit
2. Give developers example code to imitate/reuse in their own work

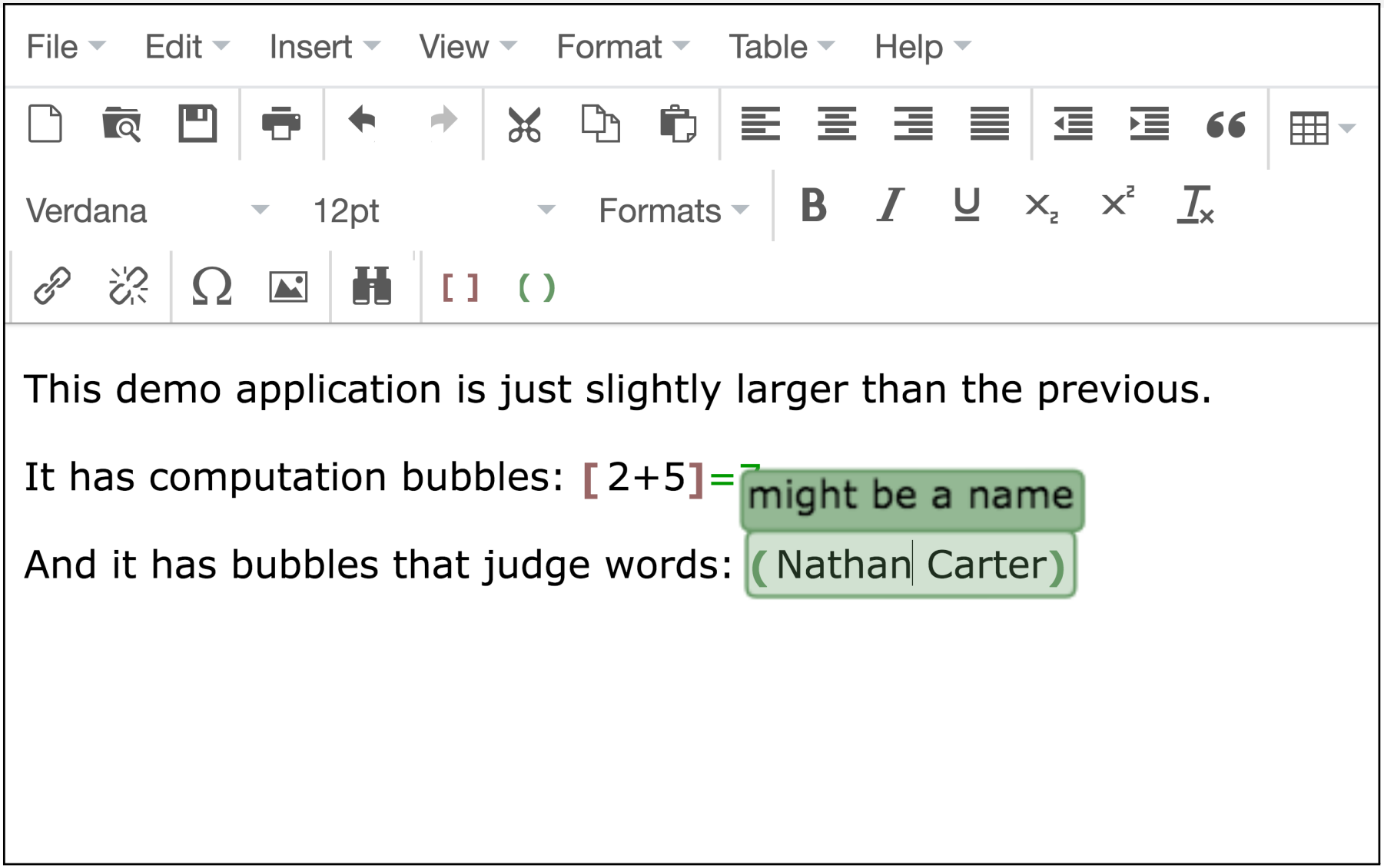
Applications:

- Simple Example — “Hello World”
- Complex Example — one step above Simple Example
- Math Evaluator — numerical evaluation of simple formulas
- OMCD Editor — creates a specific type of XML structure
- Code Editor — creates programs in a tiny language
- Lean UI — creates programs/proofs checked by Lean
- Main *Lurch* app — just beginning

Demo Application: Complex Example



Demo Application: Complex Example



Demo Applications

Two purposes:

1. Give this audience an idea of what can be done with the toolkit
2. Give developers example code to imitate/reuse in their own work

Applications:

- Simple Example — “Hello World”
- Complex Example — one step above Simple Example
- Math Evaluator — numerical evaluation of simple formulas
- OMCD Editor — creates a specific type of XML structure
- Code Editor — creates programs in a tiny language
- Lean UI — creates programs/proofs checked by Lean
- Main *Lurch* app — just beginning

Demo Applications

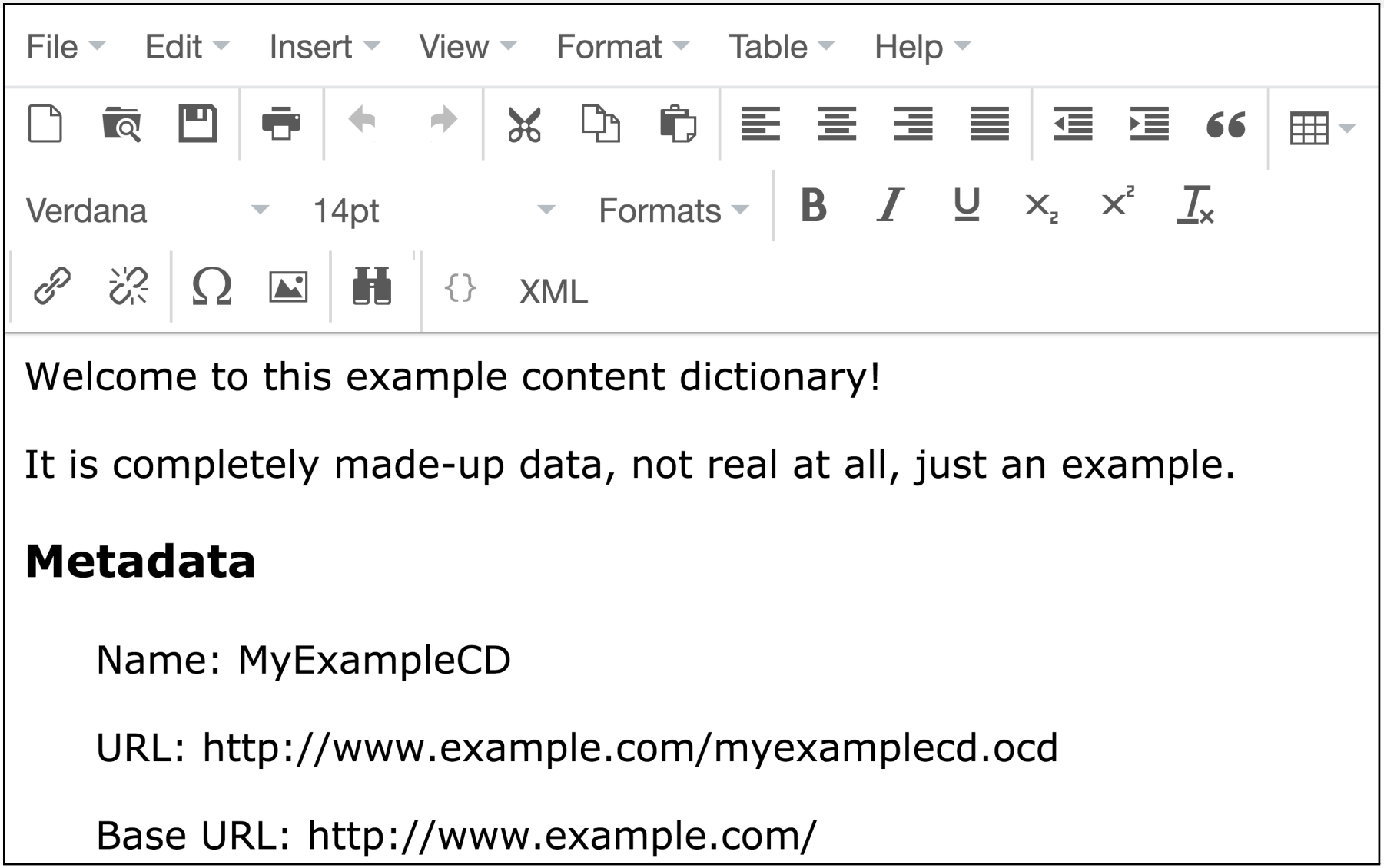
Two purposes:

1. Give this audience an idea of what can be done with the toolkit
2. Give developers example code to imitate/reuse in their own work

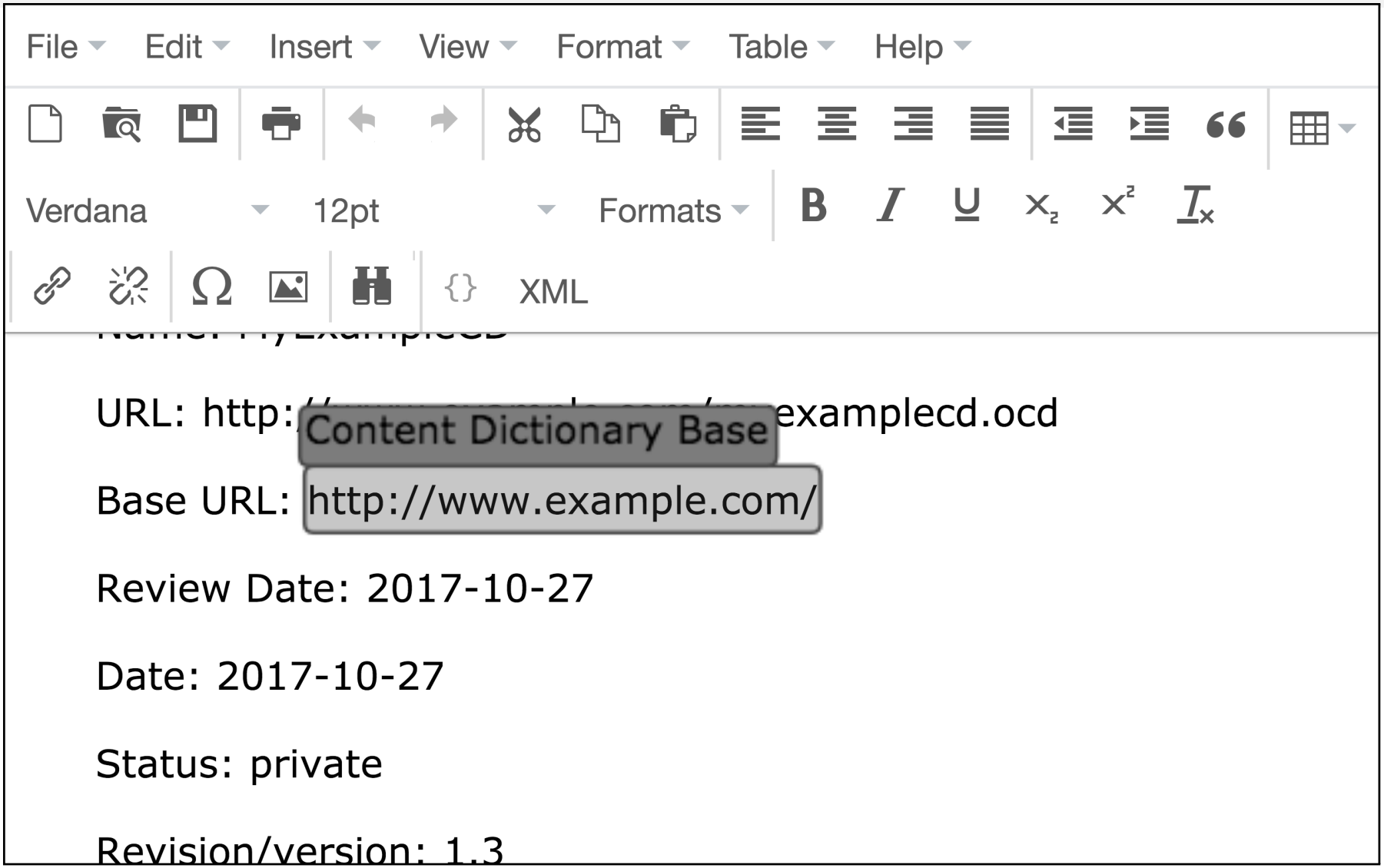
Applications:

- Simple Example — “Hello World”
- Complex Example — one step above Simple Example
- Math Evaluator — numerical evaluation of simple formulas
- OMCD Editor — creates a specific type of XML structure
- Code Editor — creates programs in a tiny language
- Lean UI — creates programs/proofs checked by Lean
- Main *Lurch* app — just beginning

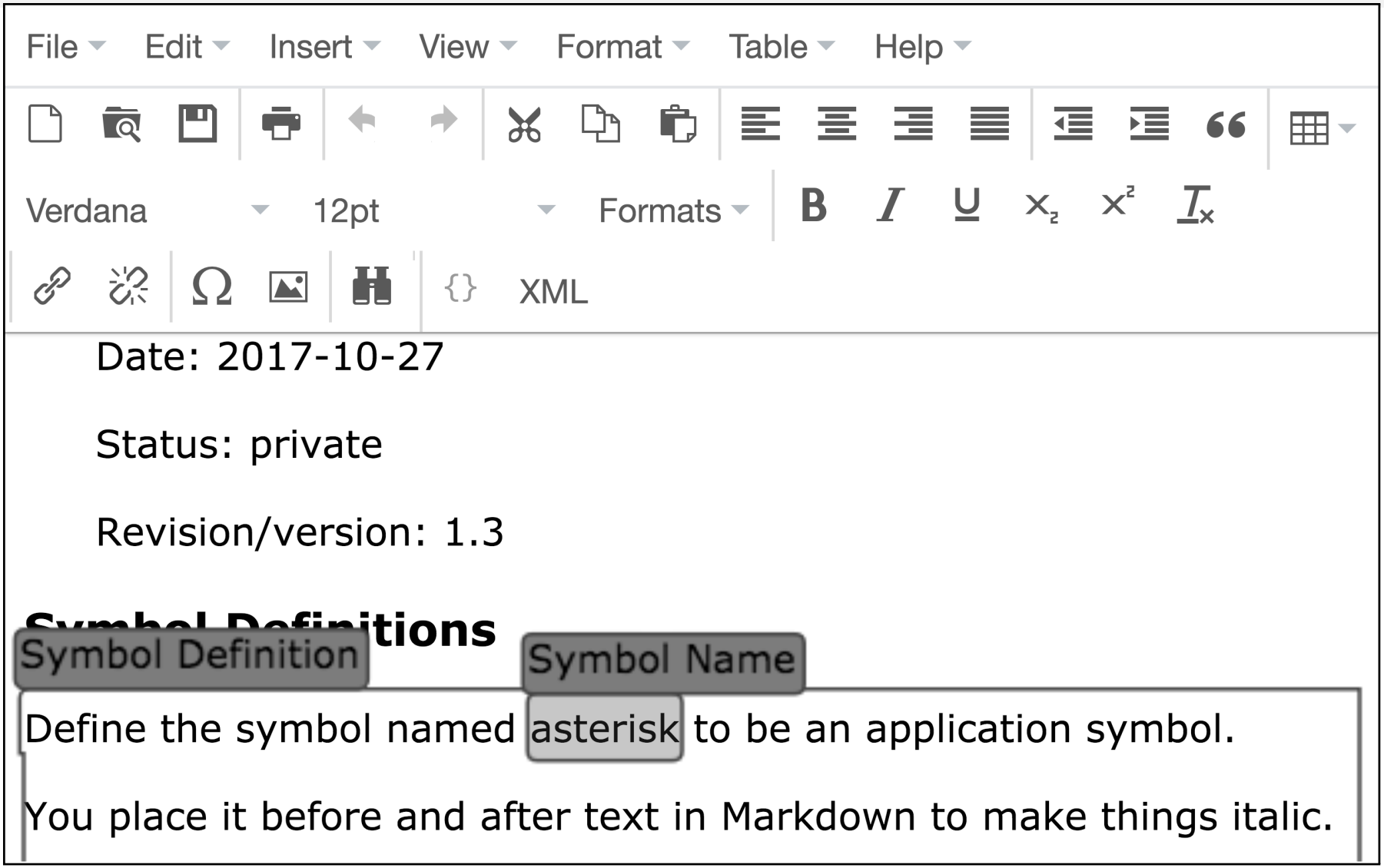
Demo Application: OpenMath CD Editor



Demo Application: OpenMath CD Editor



Demo Application: OpenMath CD Editor



Demo Applications

Two purposes:

1. Give this audience an idea of what can be done with the toolkit
2. Give developers example code to imitate/reuse in their own work

Applications:

- Simple Example — “Hello World”
- Complex Example — one step above Simple Example
- Math Evaluator — numerical evaluation of simple formulas
- OMCD Editor — creates a specific type of XML structure
- Code Editor — creates programs in a tiny language
- Lean UI — creates programs/proofs checked by Lean
- Main *Lurch* app — just beginning

Demo Application: Code Editor

File Edit Insert View Format Table Help

Verdana 12pt Formats

f(x)

Data Simple actions Input/Output Control flow Toggle code view

This application lets you create code, of a few very simple types, in English prose. It is a proof of concept that could be expanded.

Let i count from 1 to 5, and each step of the way, show the user the value of $\frac{i^2}{2}$.

Check to see if P is true.
If so, do this: Let x be the number 100.
If not, do this: Let y be the text Hello.

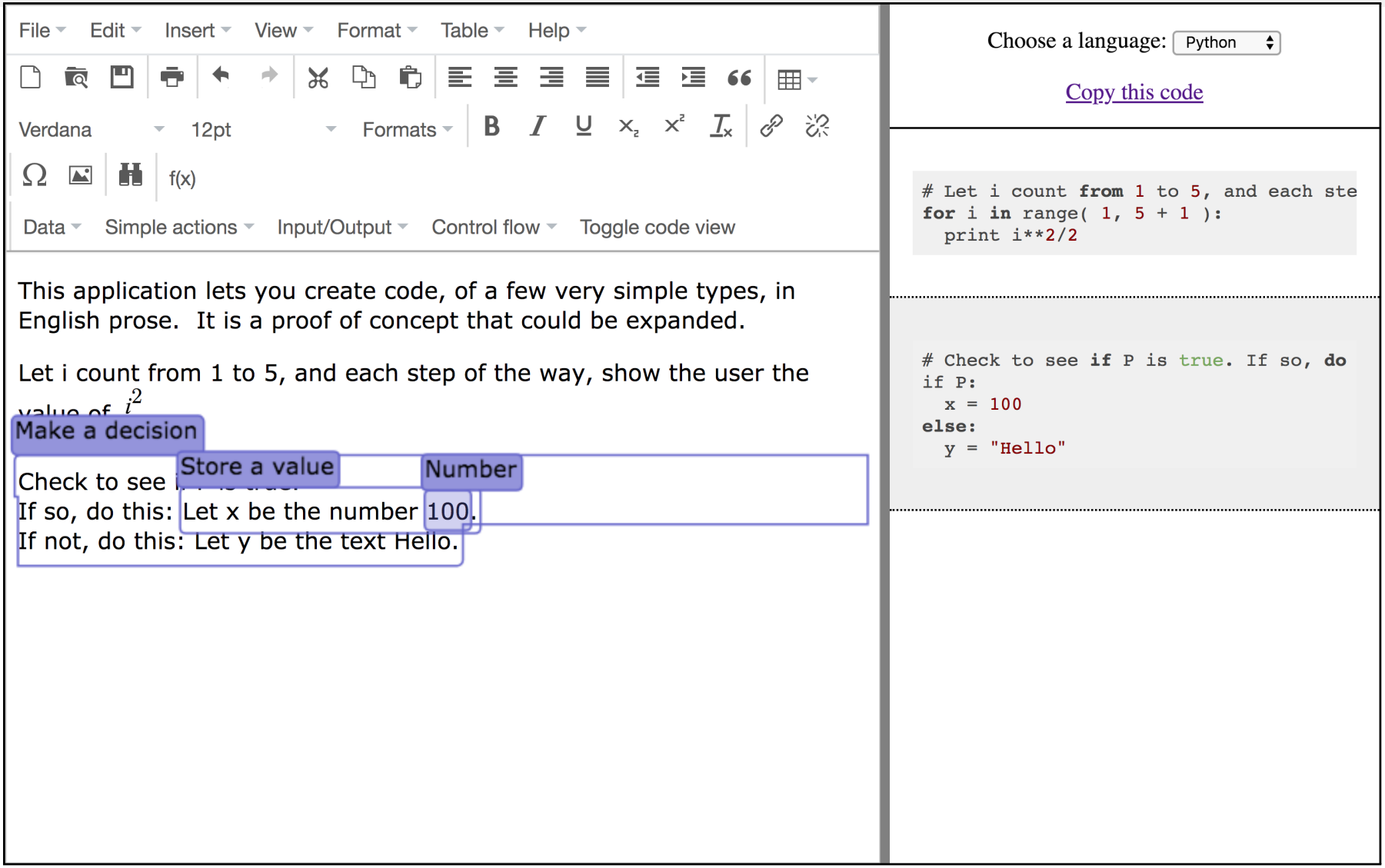
Choose a language: Python

Copy this code

```
# Let i count from 1 to 5, and each ste
for i in range( 1, 5 + 1 ):
  print i**2/2
```

```
# Check to see if P is true. If so, do
if P:
  x = 100
else:
  y = "Hello"
```

Demo Application: Code Editor



Demo Applications

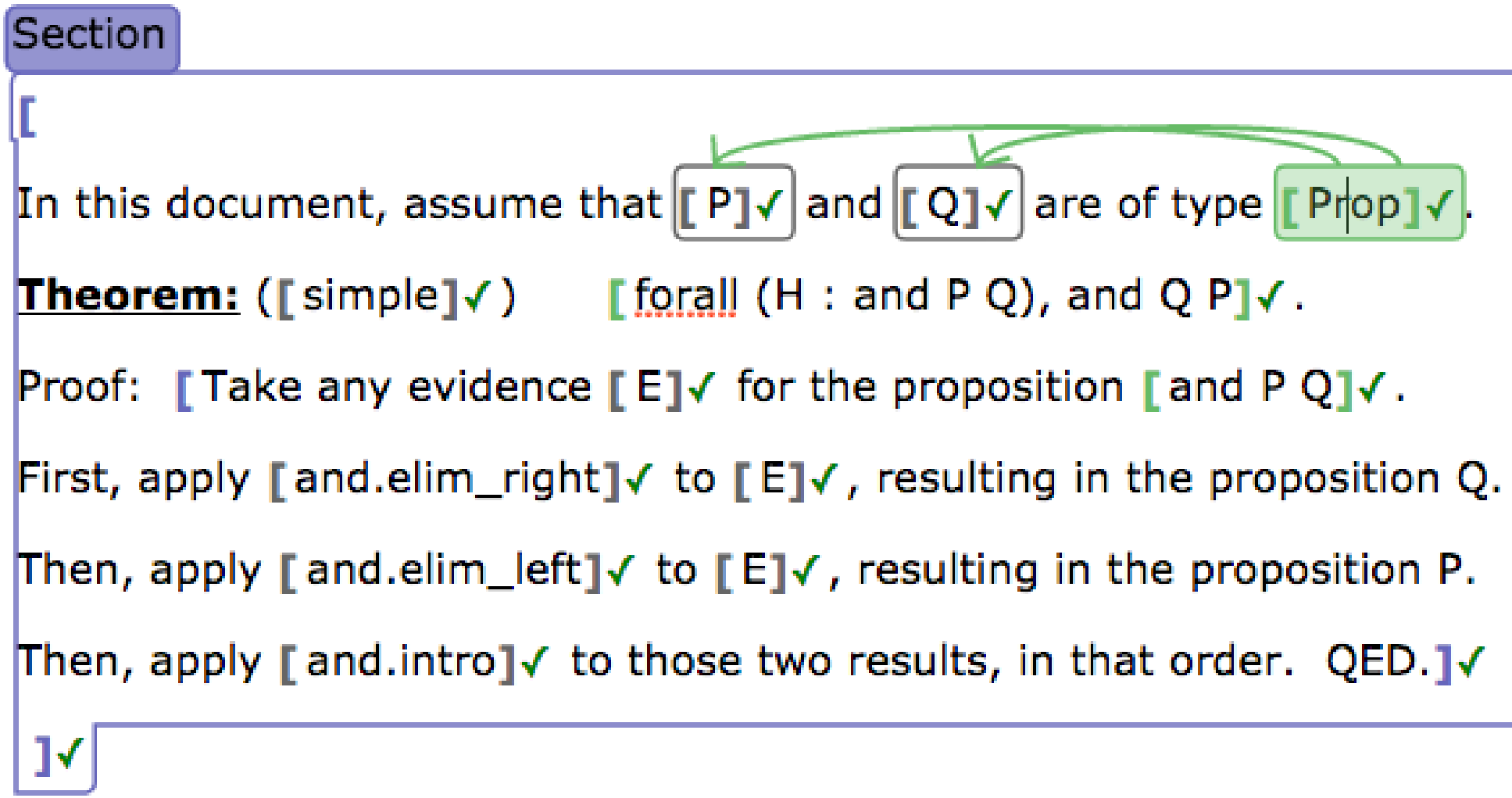
Two purposes:

1. Give this audience an idea of what can be done with the toolkit
2. Give developers example code to imitate/reuse in their own work

Applications:

- Simple Example — “Hello World”
- Complex Example — one step above Simple Example
- Math Evaluator — numerical evaluation of simple formulas
- OMCD Editor — creates a specific type of XML structure
- Code Editor — creates programs in a tiny language
- Lean UI — creates programs/proofs checked by Lean
- Main *Lurch* app — just beginning

Demo Application: Lean UI



Demo Applications

Two purposes:

1. Give this audience an idea of what can be done with the toolkit
2. Give developers example code to imitate/reuse in their own work

Applications:

- Simple Example — “Hello World”
- Complex Example — one step above Simple Example
- Math Evaluator — numerical evaluation of simple formulas
- OMCD Editor — creates a specific type of XML structure
- Code Editor — creates programs in a tiny language
- Lean UI — creates programs/proofs checked by Lean
- Main *Lurch* app — just beginning

Desktop version (current release):

`lurchmath.org`

Web development (including *Lurch Web Platform*):

`lurchmath.github.io`

Demo applications:

`lurchmath.github.io/lwp-example-simple`
`lurchmath.github.io/lwp-example-complex`
`lurchmath.github.io/lwp-example-math`
`lurchmath.github.io/lwp-example-openmath`
`lurchmath.github.io/lwp-example-sidebar`
`lurchmath.github.io/lwp-example-lean`

Using the *LWP:* Step 1. Import scripts

```
<script src='https://cdn.tinymce.com/4/
    tinymce.min.js'></script>
<script src='https://code.jquery.com/
    jquery-2.1.1.min.js'></script>
<script src='https://cdn.jsdelivr.net/gh/lurchmath/
    cloud-storage@v2.1/release/cloud-storage.js'></script>
<script src='https://unpkg.com/dropbox/dist/
    Dropbox-sdk.min.js'></script>
<script src='https://cdn.jsdelivr.net/gh/lurchmath/
    cloud-storage@v2/release/dropbox-backend.js'></script>
<script src='https://cdn.jsdelivr.net/gh/lurchmath/
    cloud-storage@v2/release/localstorage-backend.js'></scr
<script src='https://cdn.jsdelivr.net/gh/lurchmath/
    lurch@latest/release/lurch-web-platform.js'></script>
```

Using the *LWP:* Step 2. Set app name

```
<script>
setAppName("Your App Name Here");
<script>
```

Using the *LWP:* Step 3. Define group types

```
<script>

window.groupTypes = [ {
    name            : "example",
    text            : "Example Group",
    imageHTML       : "[ ]",
    openImageHTML   : "[",
    closeImageHTML  : "]",
    tagContents     : function ( group ) {
        if ( group.contentAsText() )
            return "not empty";
        else
            return "empty";
    }
} ];

<script>
```