A Language for Building Web Interfaces to Mathematical Software

Rachel Sun

Supervisors:
Professor Wolfgang Schreiner and Professor Elena Kartashova

Department:
Research Institute For Symbolic Computation (RISC) & Institute For Analysis

SUPPORTED BY FÖRDERUNG DER WISSENSCHAFTLICHEN FORSCHUNG (FWF)

Outline

- Introduction - Recap
- Preliminary Results & Timeline
- Tools
- Approach/Implementation
- Problems And Future Work
- Thesis Demo
- Conclusion

Introduction

Motivation

- A generic web application framework that enables Mathematicians to publish their solutions to the Internet.
- The solution can be written in any language or by calling the existing software.
- The framework should not be limited to a specific mathematical domain problem.

Goal

- Design and implement a framework to generate automatically web-based mathematical applications and deploy the services.
- Mathematical programmers only need to provide an interface description, workflow and necessary programs to the framework.

A More Clearer Illustration

What You HAVE:

- ✓ Mathematical solution to a particular domain
  - written in any language
  - using existing software

What You DON'T HAVE:

- ✓ Specific knowledge how to write a web application

What You WANT:

- ✓ Publish it to the Internet
- ✓ Share your knowledge to a broader audience

What You DON'T WANT:

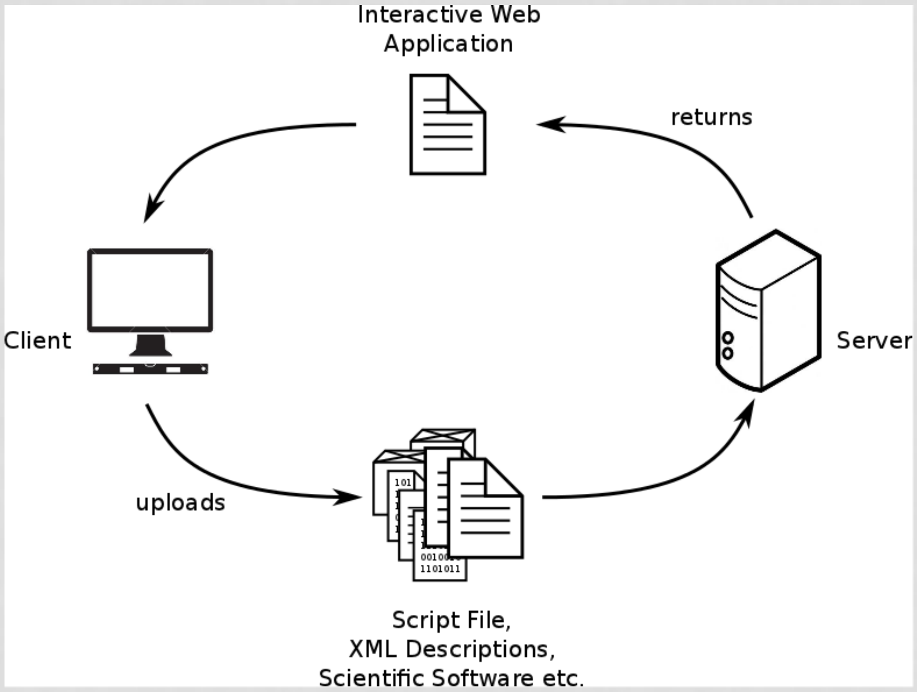
- ✓ To rewrite solution to adapt to web Technologies

A Simple Workflow

4. Web users will now be able to use the software for performing computation over a web browser.

3. It then returns either a link pointing to the web application or an error message to the client.

1. Client on his host computer uploads the necessary files for the framework to the server.



2. The server generates based on the uploaded files the interactive web application.

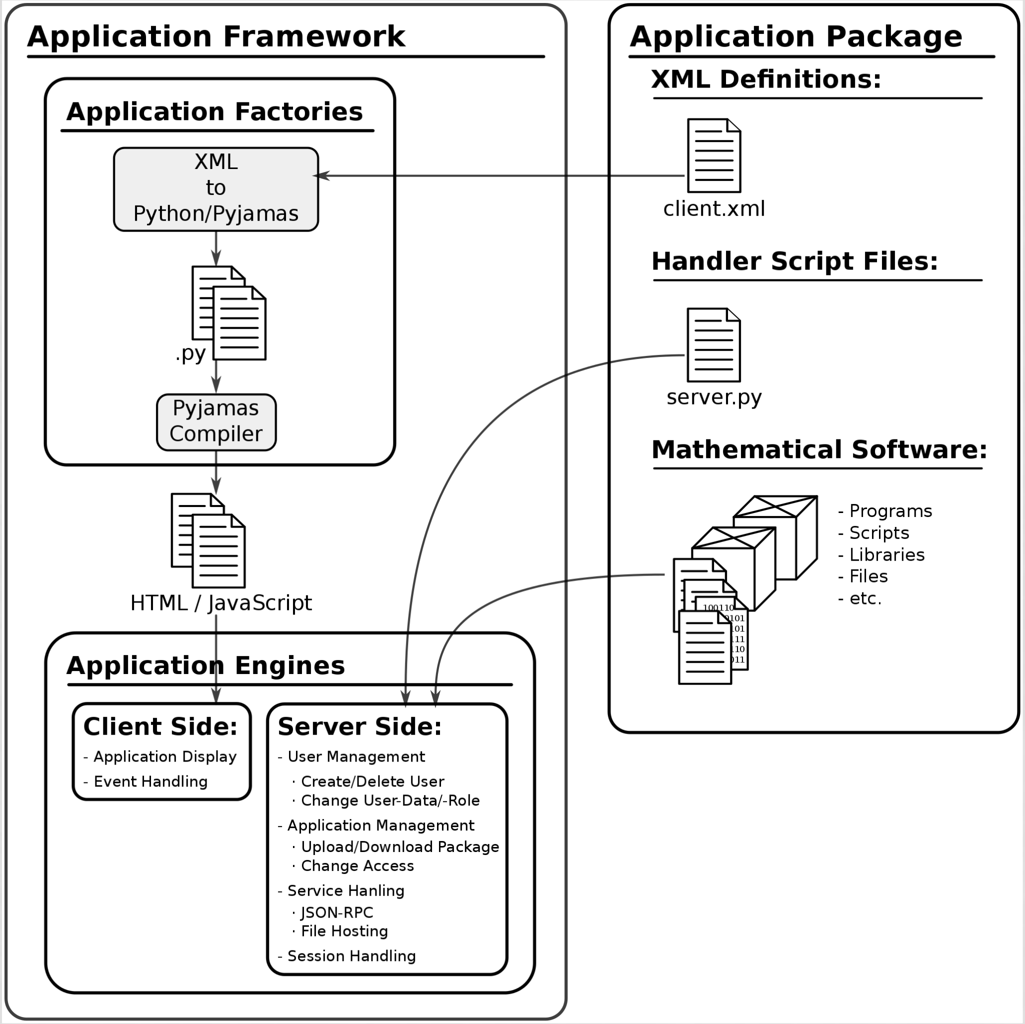
Preliminary Results & Timeline

First Semester

- ☑ Literature Reviews
- ☑ Tools Selection
- ☑ Architecture Sketch
- ☑ Prototype Development

- □ Finish Implementation
- □ Application Examples
- □ Service Testing And Evaluation
- □ System Installation on JKU Server

Framework Architecture



Application Framework

Application Factories

XML to Python/Pyjamas

.py

Pyjamas Compiler

HTML / JavaScript

Application Engines

Client Side:
- Application Display
- Event Handling

Server Side:
- User Management
  - · Create/Delete User
  - · Change User-Data/-Role
- Application Management
  - · Upload/Download Package
  - · Change Access
- Service Hanling
  - · JSON-RPC
  - · File Hosting
- Session Handling

Application Package

XML Definitions:

client.xml

Handler Script Files:

server.py

Mathematical Software:

- Programs
- Scripts
- Libraries
- Files
- etc.

Preliminary Results & Timeline

First Semester

- ☑ Literature Reviews
- ☑ Tools Selection
- ☑ Architecture Sketch
- ☑ Prototype Development

Second Semester

- ☑ Finish Implementation
- ☑ Application Examples
- ☑ Service Testing And Evaluation
- ☑ System Installation on JKU Server

Tools

Tools

First Semester:

- ✓ Eclipse Indigo 3.7
- ✓ Pydev
- ✓ Python 2.7
- ✓ XML Schema
- ✓ XML
- ✓ lxml Toolkit
- ✓ Pyjamas 0.7
- ✓ Apache Web Server 2.2.20
- ✓ JSON-RPC 2.0

Changes →

Second Semester:

- ✓ Eclipse Juno 4.2
- ✓ Pydev
- ✓ Python 2.7
- ✓ XML
- ✓ lxml Toolkit
- ✓ Pyjamas 0.7
- ✓ CherryPy 3.2.2
- ✓ JSON-RPC 2.0
- ✓ Mako Template Engine
- ✓ PostgreSQL
- ✓ Psycopg2

Tools

Pyjamas

- Free object oriented client-side web development platform.
- Write JavaScript-powered web applications in Python.
- Translates Python code to JavaScript and HTML.
- Handles all cross-browser issues for the developer.
- Necessary for package deployment.

CherryPy

- A lightweight server-side web application framework.
- Has its own built-in web server to host websites.
- Fast handling of user requests.
- Applications run on Windows, Linux and Mac OS X.
- Provides web contents and handles HTTP requests.

Tools

MAKO Template library

- Template engine for rendering HTML pages on the server-side.
- Very intuitive by using embedded Python code.
- Very fast as templates are compiled into Python byte code.

PostgreSQL

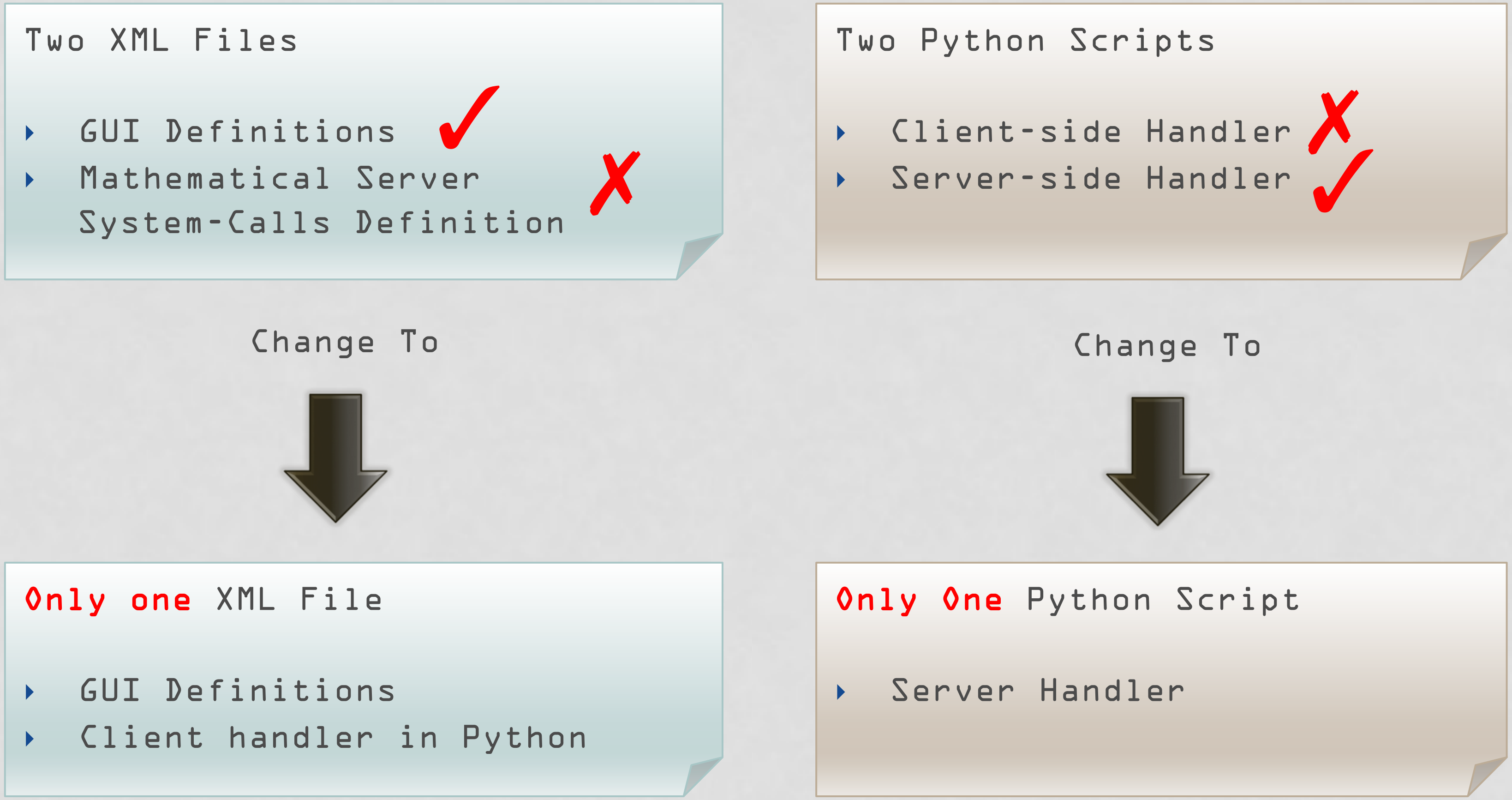
- Powerful open source object-relational database system.
- Runs on all major operating systems.
- Used for storing user, application and session data.

Psycopg2

- PostgreSQL adapter for the Python programming language.
- Fast and secure to connect to the PosgreSQL.

Approach/Implementation

Files To Write And Provide



Two XML Files

- GUI Definitions ✓
- Mathematical Server System-Calls Definition ✗

Two Python Scripts

- Client-side Handler ✗
- Server-side Handler ✓

Change To

Change To

Only one XML File

- GUI Definitions
- Client handler in Python

Only One Python Script

- Server Handler

Files To Write And Provide

Only one XML File

- GUI Definitions
- Client handler in Python

Only One Python Script

- Server Handler

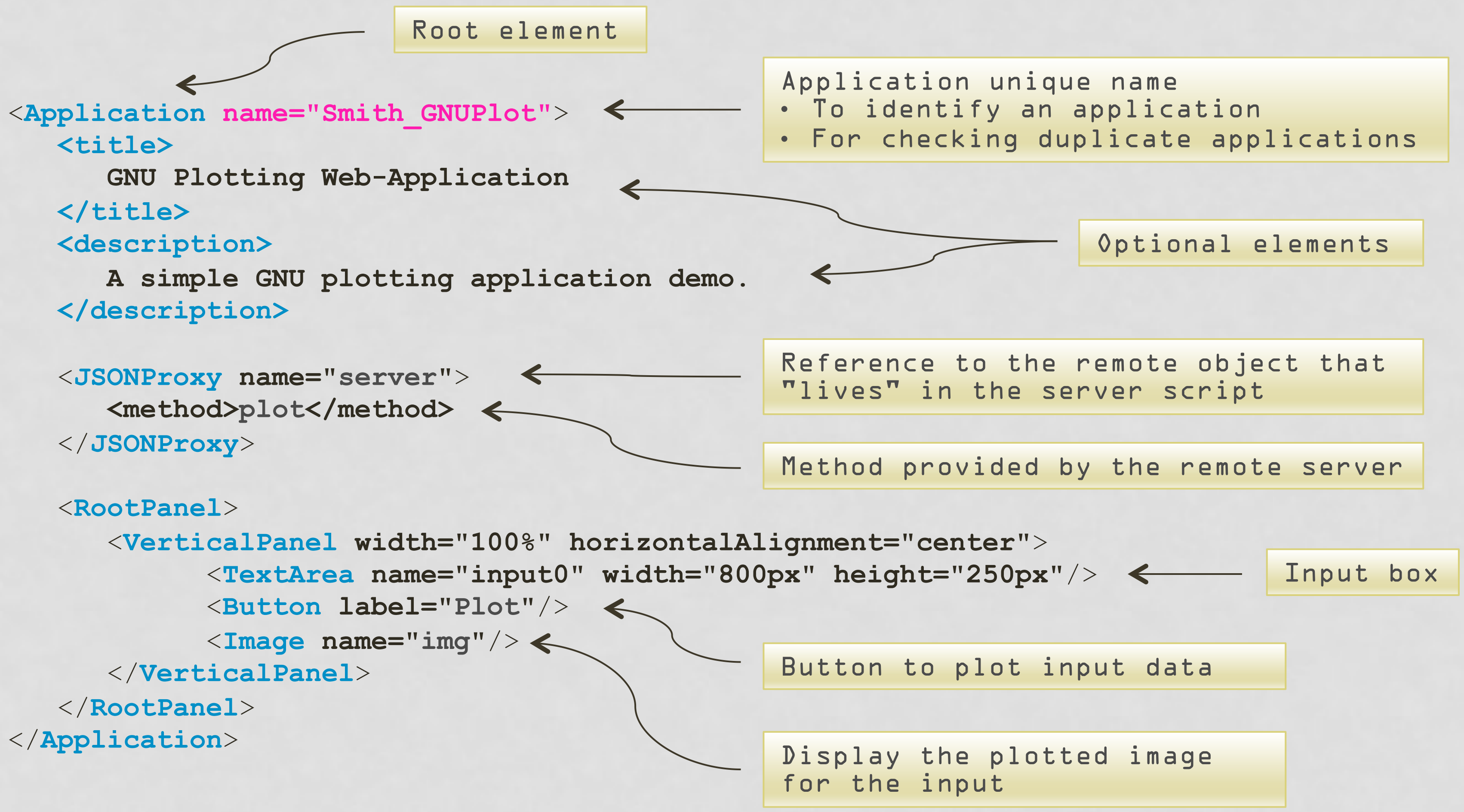
Mathematical Computation Files

- Arbitrary scripts, libraries, programs etc.
- Used to perform actual mathematical computation.

XML File: GUI Definitions

- Describe website graphical user interface (GUI) in XML:
  - How static web application should look like.
  - How panels and widgets are organized and interacted with each other.
- Define what methods are available in a remote service (JSON-RPC).
- Widgets that are currently supported:
  - Panels : Absolute Panel, Caption Panel, Dock Panel, Flow Panel, Form Panel, Horizontal Panel, Scroll Panel and Vertical Panel.
  - Widgets : Button, Check Box, File Upload, HTML, Image, Label, Radio Button, Text Area, Text Box.

Example: GNUPlot GUI Definitions



```
<Application name="Smith_GNUPlot">
   <title>
      GNU Plotting Web-Application
   </title>
   <description>
      A simple GNU plotting application demo.
   </description>

   <JSONProxy name="server">
      <method>plot</method>
   </JSONProxy>

   <RootPanel>
      <VerticalPanel width="100%" horizontalAlignment="center">
            <TextArea name="input0" width="800px" height="250px"/>
            <Button label="Plot"/>
            <Image name="img"/>
      </VerticalPanel>
   </RootPanel>
</Application>
```

Root element

Application unique name
- To identify an application
- For checking duplicate applications

Optional elements

Reference to the remote object that "lives" in the server script

Method provided by the remote server

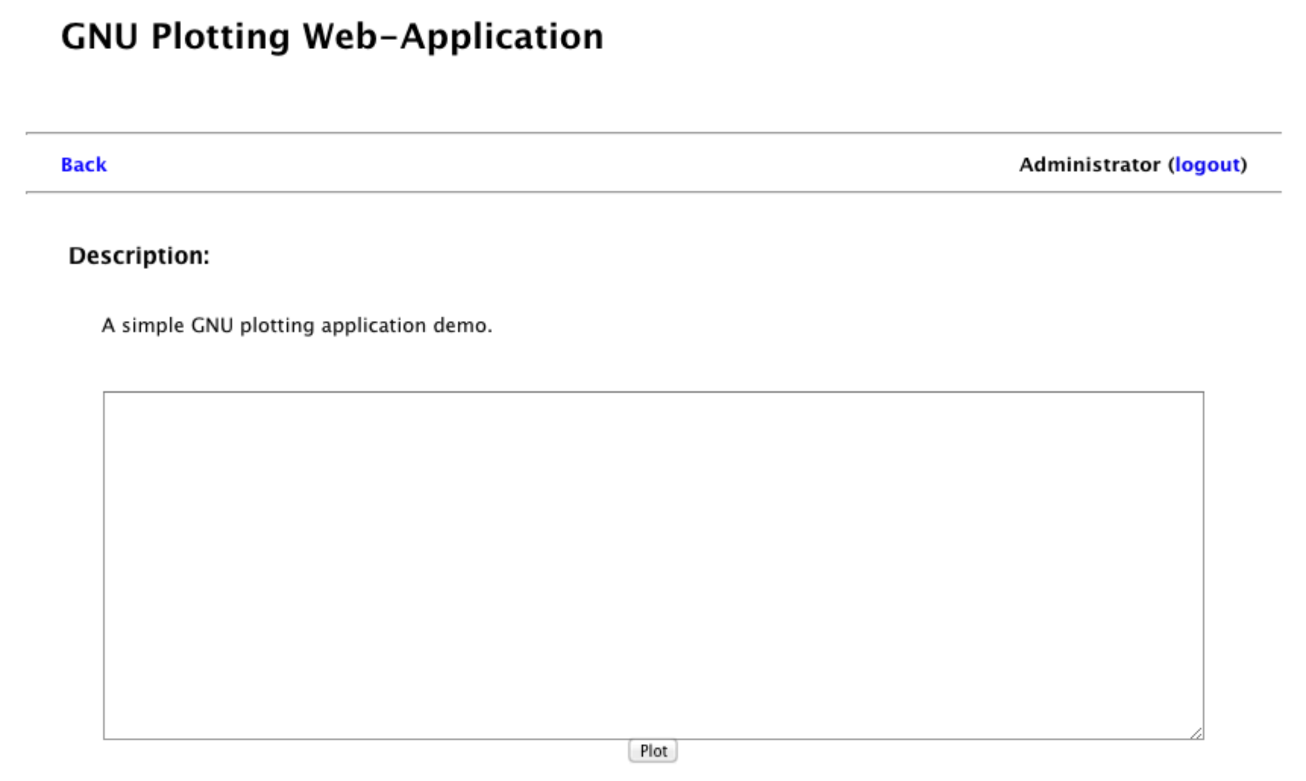
Input box

Button to plot input data

Display the plotted image for the input

Example: A GNU Plotting GUI

A GNU Plotting Web Application Graphical User Interface (GUI):



GNU Plotting Web-Application

Back

Administrator (logout)

Description:

A simple GNU plotting application demo.

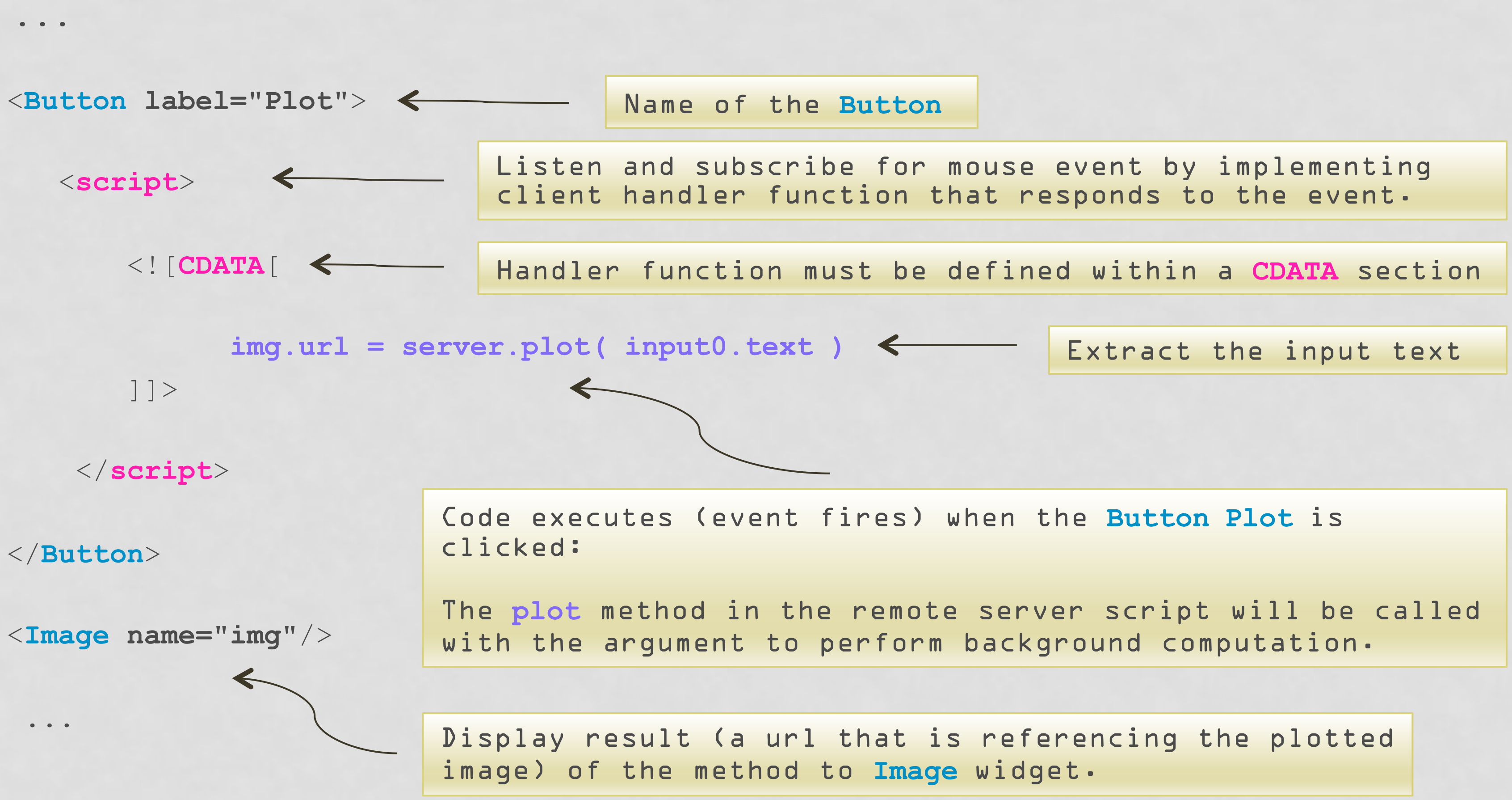
XML File: Client Handler

- Describe workflow in Python:
  - How to handle user interaction with the GUI objects when events are fired on the web browser.
  - Acts as an event handler that listens for a 'change' event on the widgets.
- A client handler function for a widget can be defined in **two ways:**
  - Inside the CDATA section of a script element of a **widget**.

    Benefit: Particularly useful when you have only one Button widget.
  - Inside the CDATA section of a script element within **root Application** with a function name. Then define the function name in the listener attribute of a Button widget.

    Benefit: A single handler can be shared among many widgets.

Method 1: Client Handler Definition



```
...

<Button label="Plot">

   <script>

       <![CDATA[

             img.url = server.plot( input0.text )
       ]]>

     </script>

</Button>

<Image name="img"/>

 ...
```

Name of the Button

Listen and subscribe for mouse event by implementing client handler function that responds to the event.

Handler function must be defined within a CDATA section

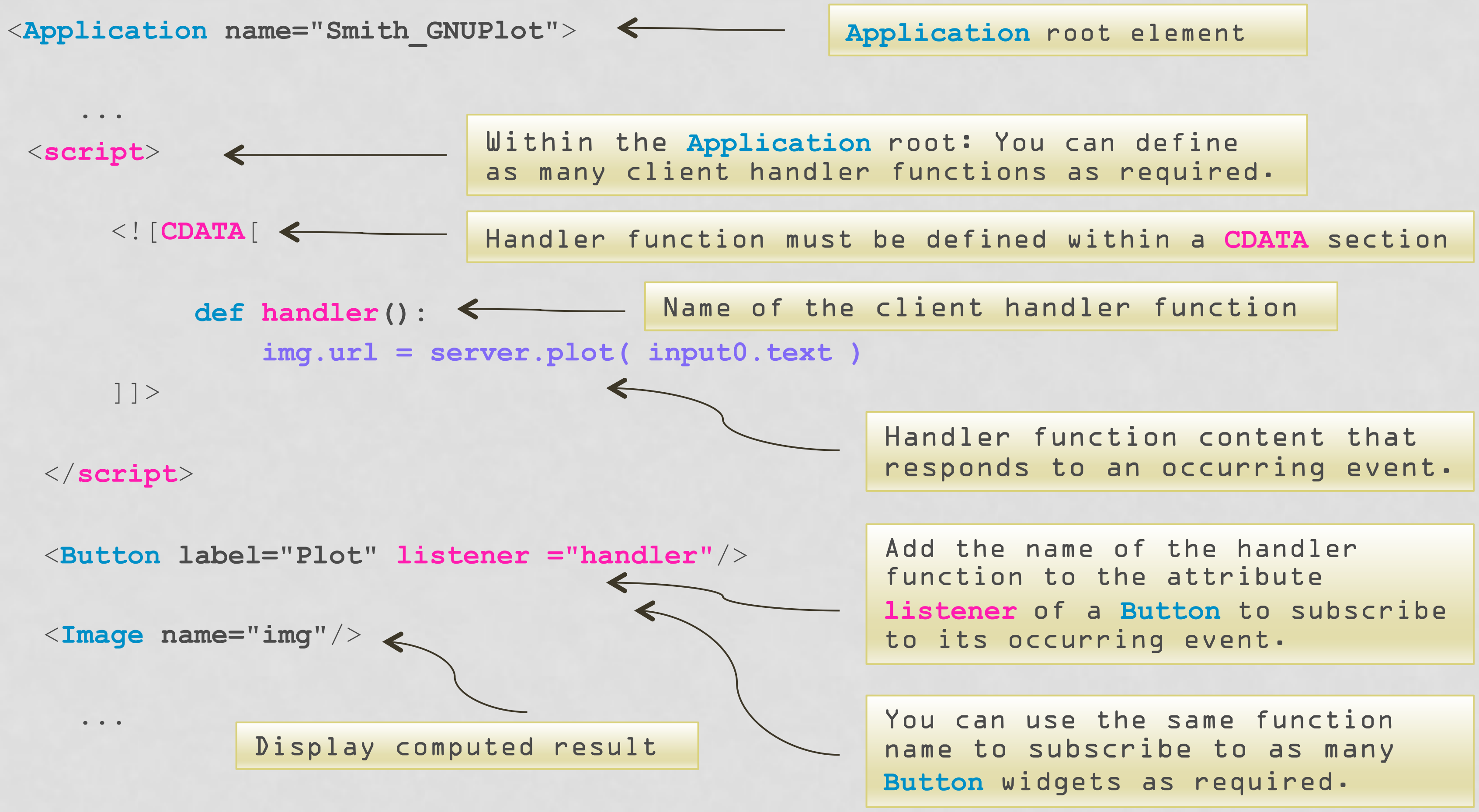
Extract the input text

Code executes (event fires) when the Button Plot is clicked:

The plot method in the remote server script will be called with the argument to perform background computation.

Display result (a url that is referencing the plotted image) of the method to Image widget.

Method 2: Client Handler Definition



```
<Application name="Smith_GNUPlot">

    ...
  <script>

      <![CDATA[

          def handler():
              img.url = server.plot( input0.text )
      ]]>

   </script>

   <Button label="Plot" listener ="handler"/>

   <Image name="img"/>

    ...
```

Application root element

Within the Application root: You can define as many client handler functions as required.

Handler function must be defined within a CDATA section

Name of the client handler function

Handler function content that responds to an occurring event.

Add the name of the handler function to the attribute listener of a Button to subscribe to its occurring event.

You can use the same function name to subscribe to as many Button widgets as required.

Display computed result

Python Script: Server Handler

- A pure Python module.
- Describe how to handle user requests on the server by defining functions.
  - How to call the mathematical software in the background to perform actual computation.
- All methods implemented are exposed as JSON-RPCs.
- User can invoke any method defined in the server handler script from the XML client application.

Example: GNUPlot Server Handler

Use the function name to expose the function as JSON-RPC in XML client

User-entered input is the parameter to the function

```
def plot( src ):

    im_file = "%016x.png" % ( time() * 1000000 )

    gp_head = """set terminal png;\n""" \
              """set output '%s';\n""" % im_file

    gp_proc = Popen( "gnuplot", shell=True, stdin=PIPE,
                     stdout=PIPE, stderr=STDOUT )

    stdout  = gp_proc.communicate( gp_head + src )[0]


    if gp_proc.returncode:
        raise Exception( stdout )
    return im_file
```

Generate unique filenames for output image plots.

Define a header for the GNUPlot input to write the result into output file

Create a sub-process instance of GNUPlot application

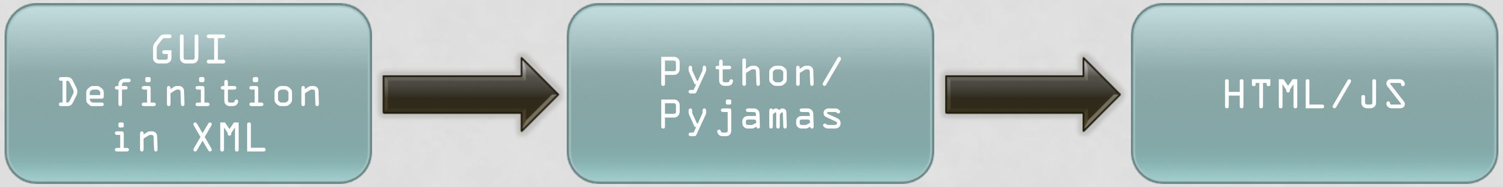
Interact with GNUPlot process:

- Send header and input text data to **stdin**.
- Method call returns data from **stdout** and **stderr**.

Return the URL of the plotted image if no error occurs

GUI Definition (XML) -> HTML/JS

- XML needs to be transpiled into HTML/JS:
  - Something every browsers should understand.
- But first transpile it to an intermediate format: Python/Pyjamas
- Finally we transpile the Python/Pyjamas source to HTML/JS.
  - This transpilation will be done by the Pyjamas-Framework: pyjsbuild



GUI Definition in XML

Python/ Pyjamas

HTML/JS

Example:

GUI Definition(XML)-> Python/Pyjamas

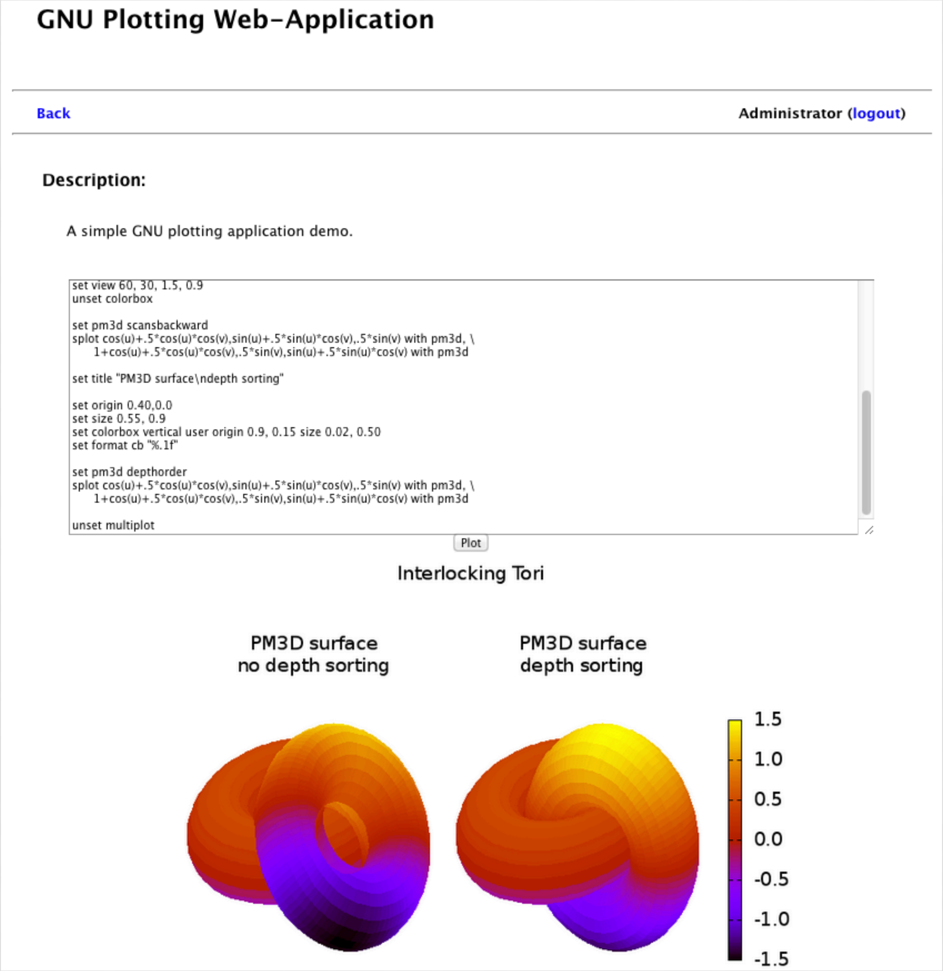
GUI Definition

```
...

...

<JSONProxy name="server">
  <method>plot</method>
</JSONProxy>

<RootPanel>
  <VerticalPanel width="100%"
      horizontalAlignment="center">
    <TextArea name="input0" width="800px"
      height="250px"/>
    <Button label="Plot">
      <script>
      <![CDATA[
        img.url = server.plot( input0.text )
      ]]>
      </script>
    </Button>
    <Image name="img"/>
  </VerticalPanel>
</RootPanel>

...
```

Intermediate Format

```
class GNUPlot(object):

  def __init__(self):
    self.server = ui.wrappers.produce('JSONProxy', 'services',
['plot'])
    self._RootPanel_0 = ui.wrappers.produce('RootPanel')
    self._VerticalPanel_0 = ui.wrappers.produce('VerticalPanel',
      HorizontalAlignment=HasAlignment.ALIGN_CENTER, Width='100%')
    self.input0 = ui.wrappers.produce('TextArea', Width='800px',
      Height='250px')
    self._Button_0 = ui.wrappers.produce('Button', html='Plot',
      listener=self._Button_0_listener_)
    self.img = ui.wrappers.produce('Image')

    global server,input0,img
    server = self.server
    input0 = self.input0
    img    = self.img

  def onModuleLoad(self):
    self._VerticalPanel_0._widget_.add(self.input0._widget_)
    self._VerticalPanel_0._widget_.add(self._Button_0._widget_)
    self._VerticalPanel_0._widget_.add(self.img._widget_)
    self._RootPanel_0._widget_.add(self._VerticalPanel_0._widget_)

  def _Button_0_listener_(self, sender):
    img.url = server.plot( input0.text )
```

Example: A GNU Plotting Web Application



GNU Plotting Web-Application

Description:

A simple GNU plotting application demo.

```
set view 60, 30, 1.5, 0.9
unset colorbox

set pm3d scansbackward
splot cos(u)+.5*cos(u)*cos(v),sin(u)+.5*sin(u)*cos(v),.5*sin(v) with pm3d, \
     1+cos(u)+.5*cos(u)*cos(v),.5*sin(v),sin(u)+.5*sin(u)*cos(v) with pm3d

set title "PM3D surface\ndepth sorting"

set origin 0.40,0.0
set size 0.55, 0.9
set colorbox vertical user origin 0.9, 0.15 size 0.02, 0.50
set format cb "%.1f"

set pm3d depthorder
splot cos(u)+.5*cos(u)*cos(v),sin(u)+.5*sin(u)*cos(v),.5*sin(v) with pm3d, \
     1+cos(u)+.5*cos(u)*cos(v),.5*sin(v),sin(u)+.5*sin(u)*cos(v) with pm3d

unset multiplot
```

Plugin Ability of Widgets and Panels

- Widgets and panels are implemented as **Plugins** in the WebMaths framework.

  Benefit: Easy to extend the framework with new widgets or panels by implementing Plugins.

- Plugins consist of **Generators** and **Wrappers**:
  - Generators will generate Python/Pyjamas source code fragments for widgets and panels.
  - Wrappers will wrap Pyjamas widgets and panels so that the user can access them in an easier way within the XML client handler script.

  Benefit:

  - User is entirely independent of the official Pyjamas API.
  - In case Pyjamas API changed, WebMaths API shall remain untouched as regards their content and their validity.
  - User applications shall also remain intact and continuously function without causing any changes.

Server Application

- Implementation is based on the CherryPy framework.
- Is a server itself that hosts mathematical web applications and provides additional services (e.g. JSON-RPC for the server handler)
- Offers **three core features** in management of:
  - User/Account
  - Session
  - Application

User/Account Management

- Provides **three types** of user accounts:
  - Administrator
  - User
  - Guest

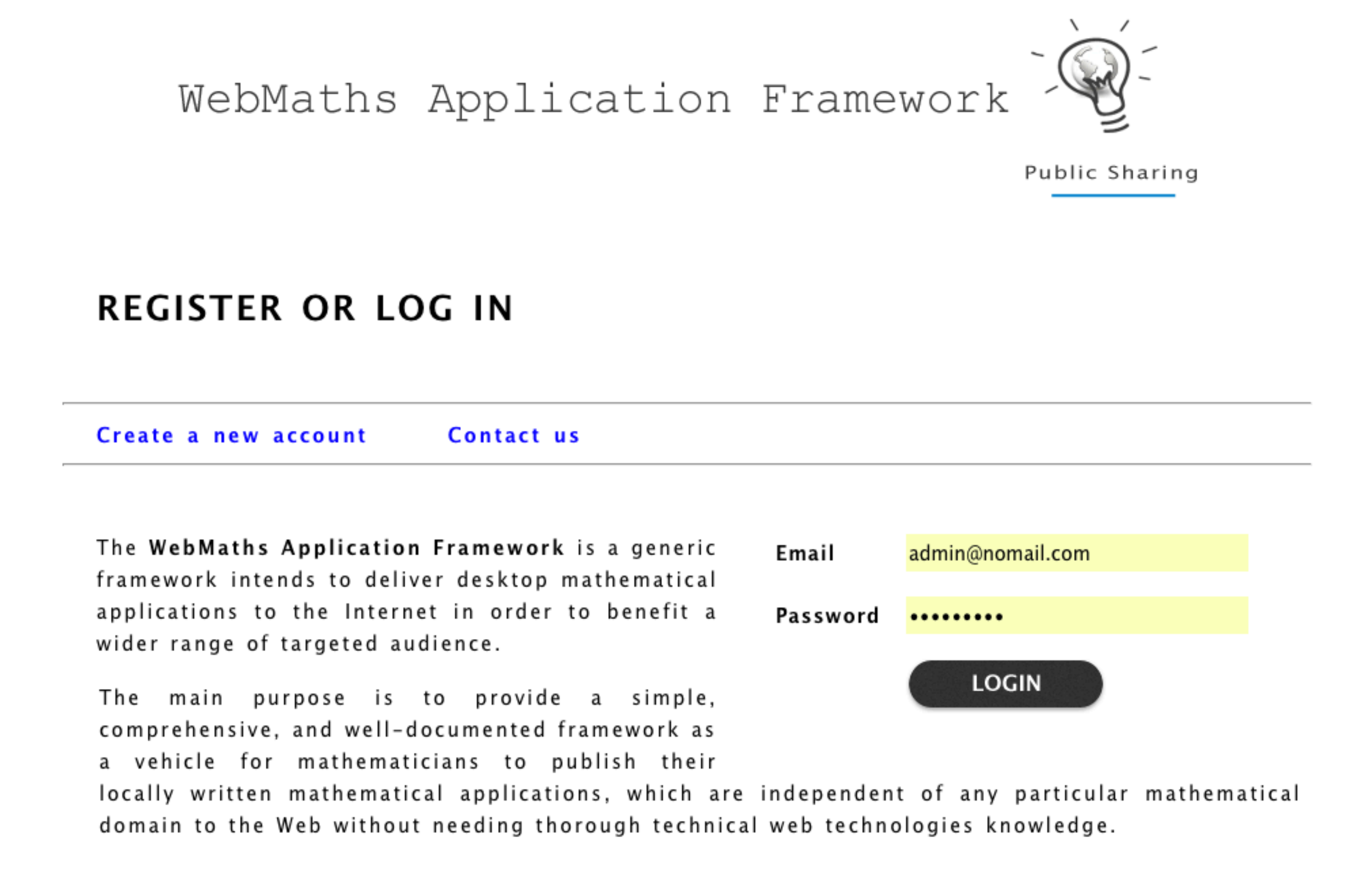
Administrator Account:

- Create new Users
- Delete existing Users
- Edit profiles of Users by changing their
  - Name
  - Password
  - Roles (eg. Administrator, User, Guest)

Standard User and Guest Account:

- Change name
- Change password

WebMaths Framework Login Page



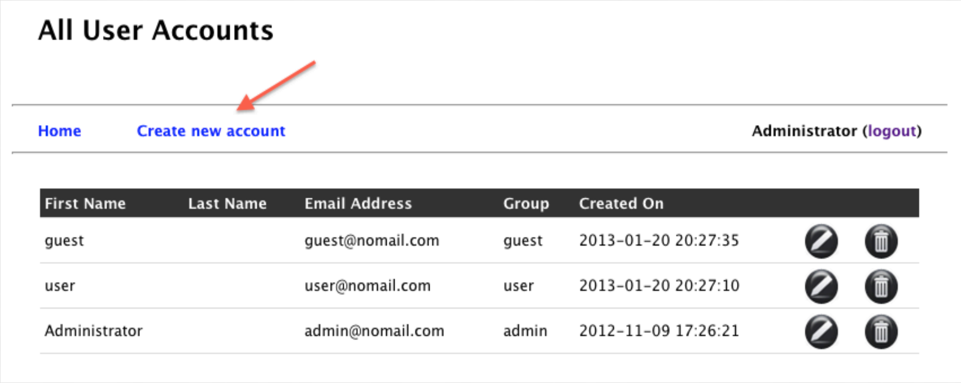
REGISTER OR LOG IN

The **WebMaths Application Framework** is a generic framework intends to deliver desktop mathematical applications to the Internet in order to benefit a wider range of targeted audience.

The main purpose is to provide a simple, comprehensive, and well-documented framework as a vehicle for mathematicians to publish their locally written mathematical applications, which are independent of any particular mathematical domain to the Web without needing thorough technical web technologies knowledge.

Email	admin@nomail.com
Password	•••••••••

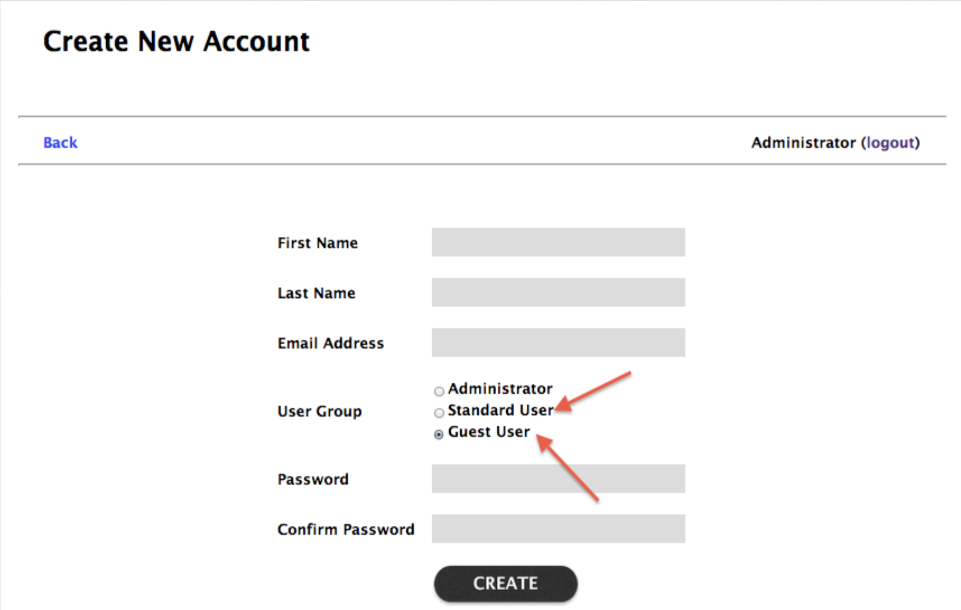
Administrator Account: Create New User Account



All User Accounts

Home   Create new account   Administrator (logout)

First Name	Last Name	Email Address	Group	Created On		
guest		guest@nomail.com	guest	2013-01-20 20:27:35	✎	🗑
user		user@nomail.com	user	2013-01-20 20:27:10	✎	🗑
Administrator		admin@nomail.com	admin	2012-11-09 17:26:21	✎	🗑

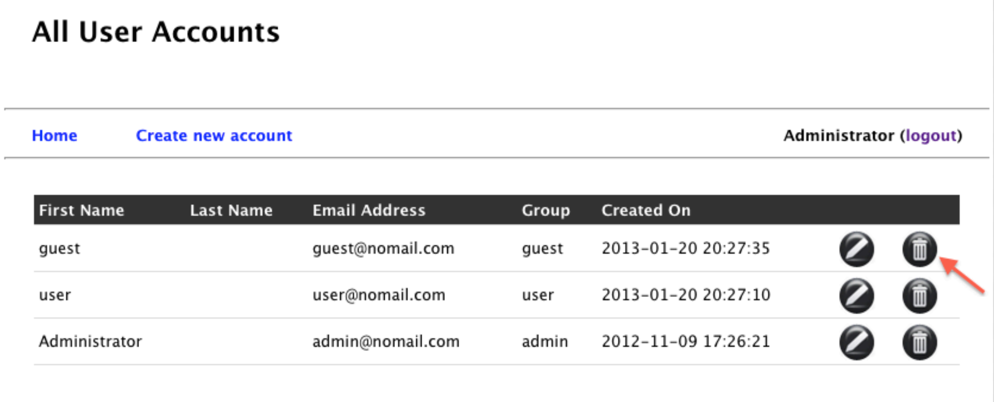


Create New Account

Back   Administrator (logout)

First Name	
Last Name	
Email Address	
User Group	○ Administrator ○ Standard User ● Guest User
Password	
Confirm Password	

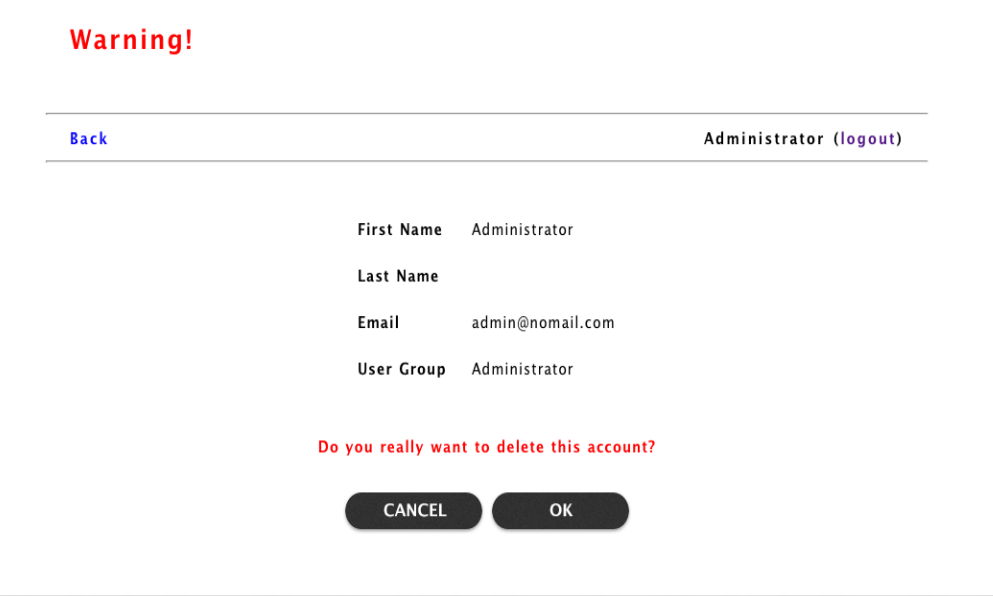
Administrator Account:
Delete Existing User Account



All User Accounts

Home | Create new account | Administrator (logout)

First Name	Last Name	Email Address	Group	Created On
guest		guest@nomail.com	guest	2013-01-20 20:27:35
user		user@nomail.com	user	2013-01-20 20:27:10
Administrator		admin@nomail.com	admin	2012-11-09 17:26:21



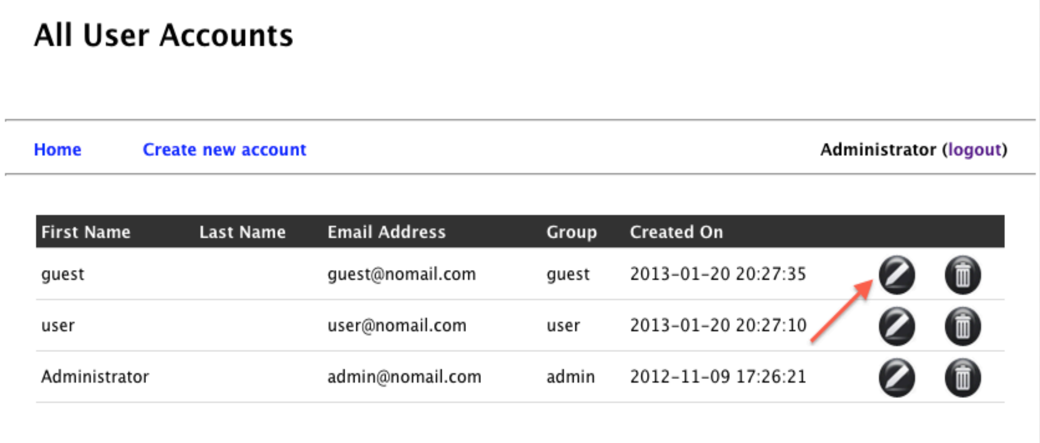
Warning!

Back | Administrator (logout)

First Name	Administrator
Last Name	
Email	admin@nomail.com
User Group	Administrator

Do you really want to delete this account?

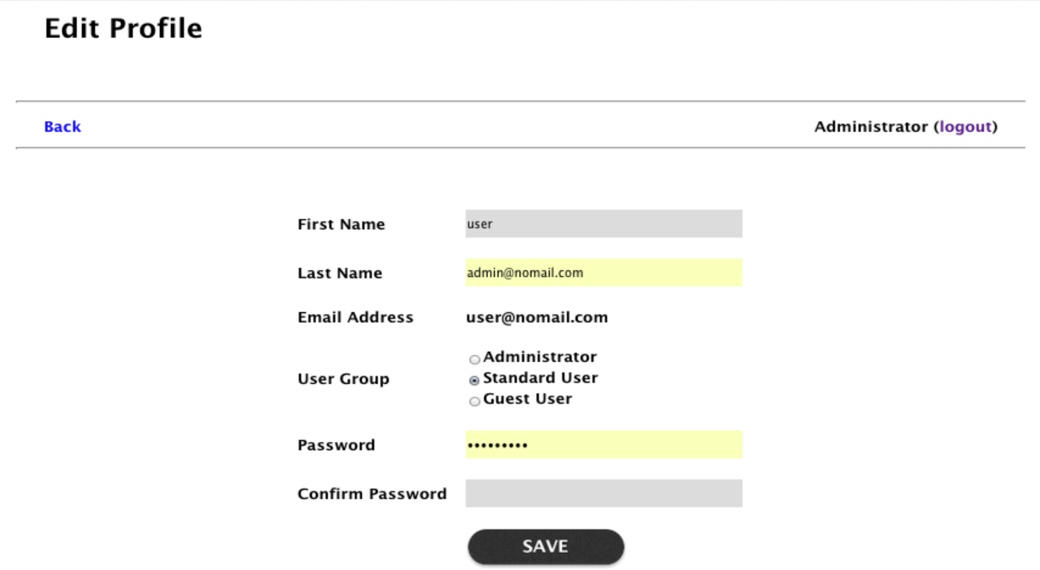
Administrator Account: Edit All Users Profile



All User Accounts

Home   Create new account   Administrator (logout)

First Name	Last Name	Email Address	Group	Created On
guest		guest@nomail.com	guest	2013-01-20 20:27:35
user		user@nomail.com	user	2013-01-20 20:27:10
Administrator		admin@nomail.com	admin	2012-11-09 17:26:21

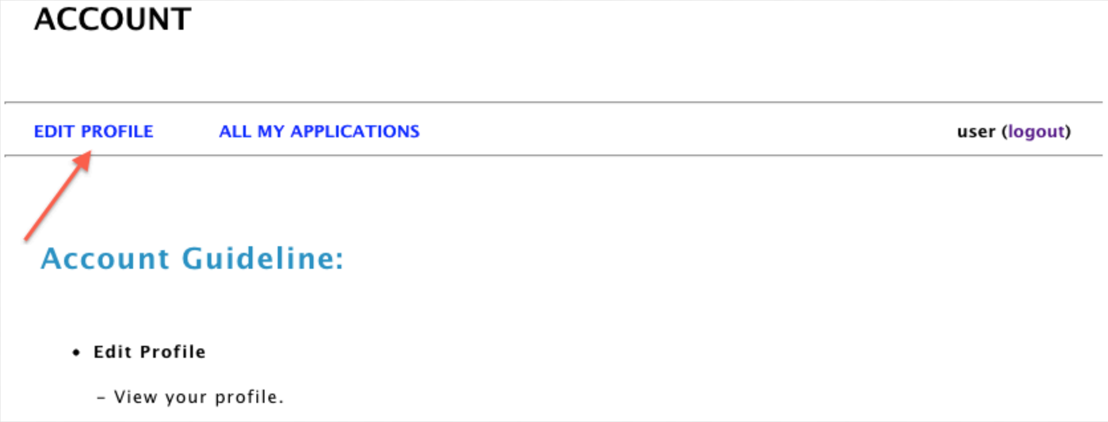


Edit Profile

Back   Administrator (logout)

First Name	user
Last Name	admin@nomail.com
Email Address	user@nomail.com
User Group	○ Administrator ● Standard User ○ Guest User
Password	•••••••••
Confirm Password	

Standard User/Guest Account: Edit User Profile

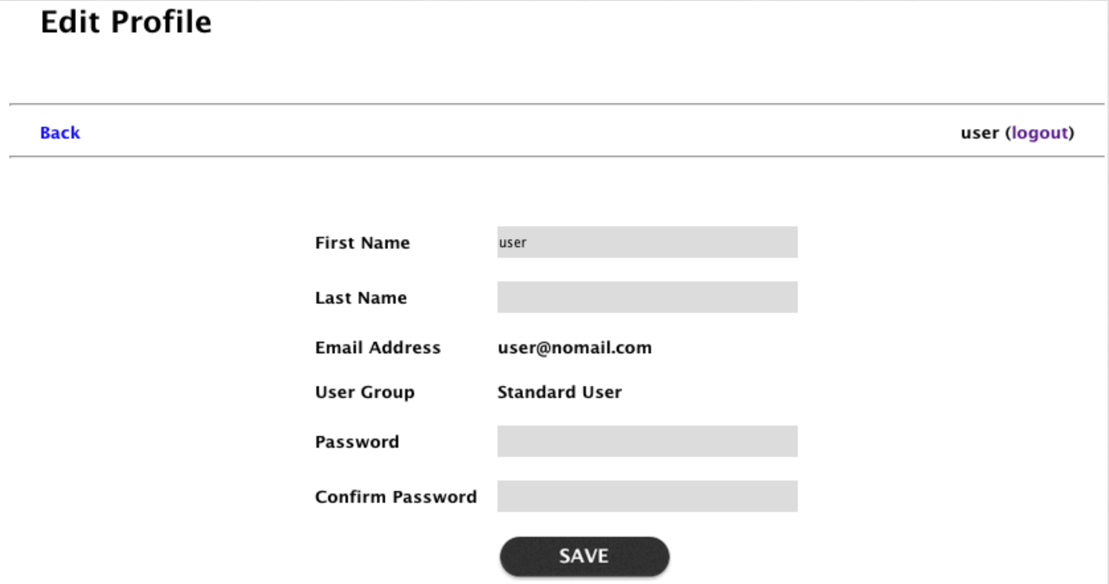


ACCOUNT

Account Guideline:

- Edit Profile

  – View your profile.



Edit Profile

First Name	user
Last Name	
Email Address	user@nomail.com
User Group	Standard User
Password	
Confirm Password	

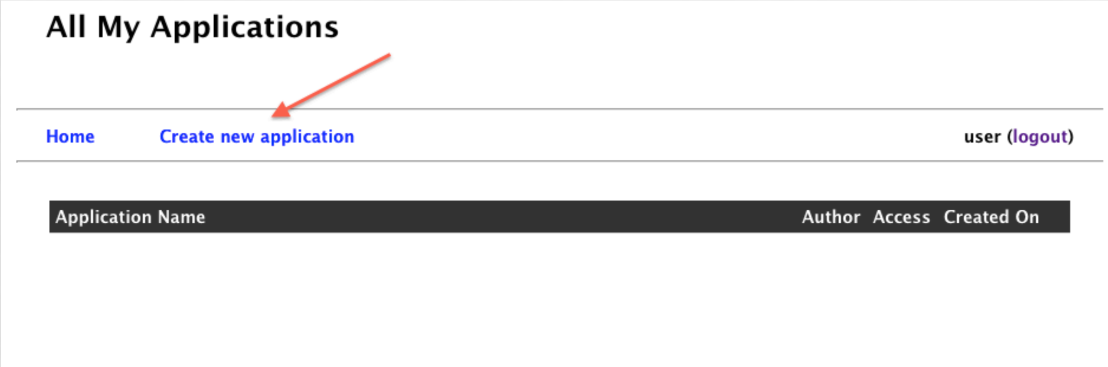
Session/Login Management

- Every user will get an unique session ID, no matter whether they are visitors or registered users.
- A Session is used to store data for a particular user:
  - Every user has its own session data.
- If a web application's server handler function stored files on the server (e.g. plot results in the GNUPlot example):
  - These files will be stored in an unique application session directory.
  - They will only exist as long as the user's session ID is valid.

Application Management

- For **Administrator and Standard User Account:**
  - View a list of all uploaded applications
  - Upload/deploy applications
  - Delete applications
  - Download the package of an application
  - Change the visibility of an application (e.g. private, users, public)
- For **all account types (Administrator, Standard User, Guest):**
  - View and access to other user-shared applications

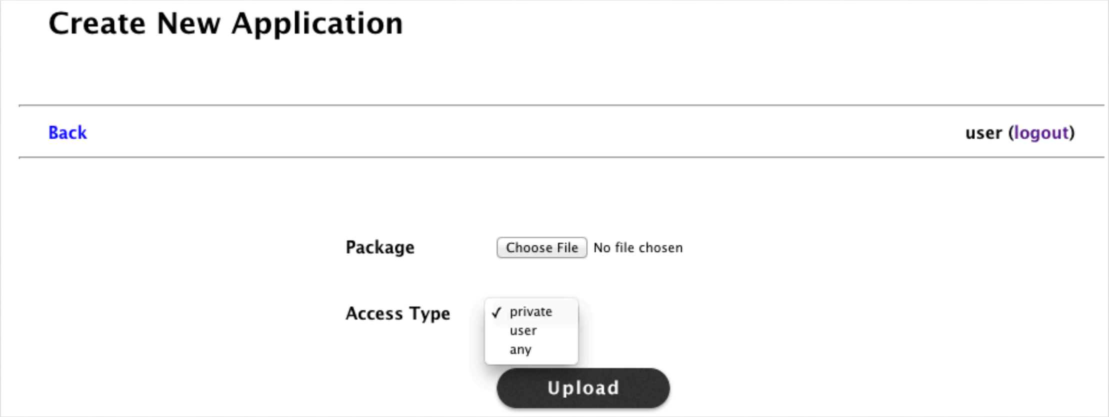
User Account:
Upload New Applications



All My Applications

Home Create new application user (logout)

Application Name	Author	Access	Created On



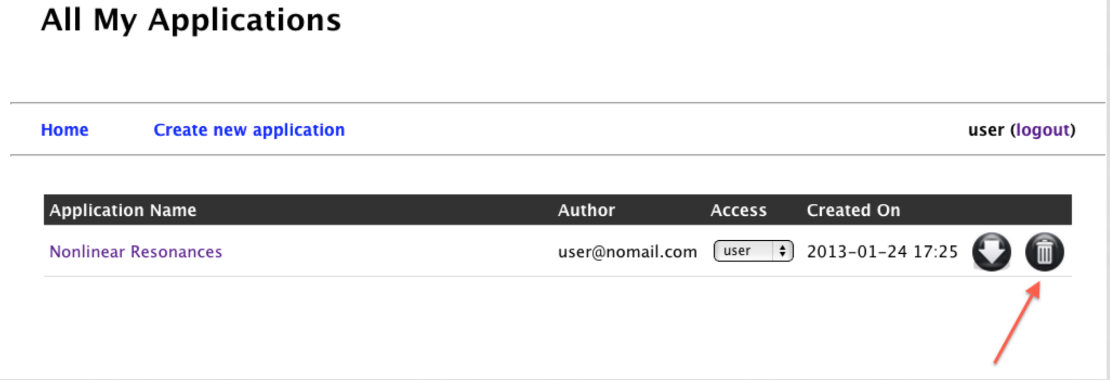
Create New Application

Back user (logout)

Package: Choose File No file chosen

Access Type: ✓ private
user
any

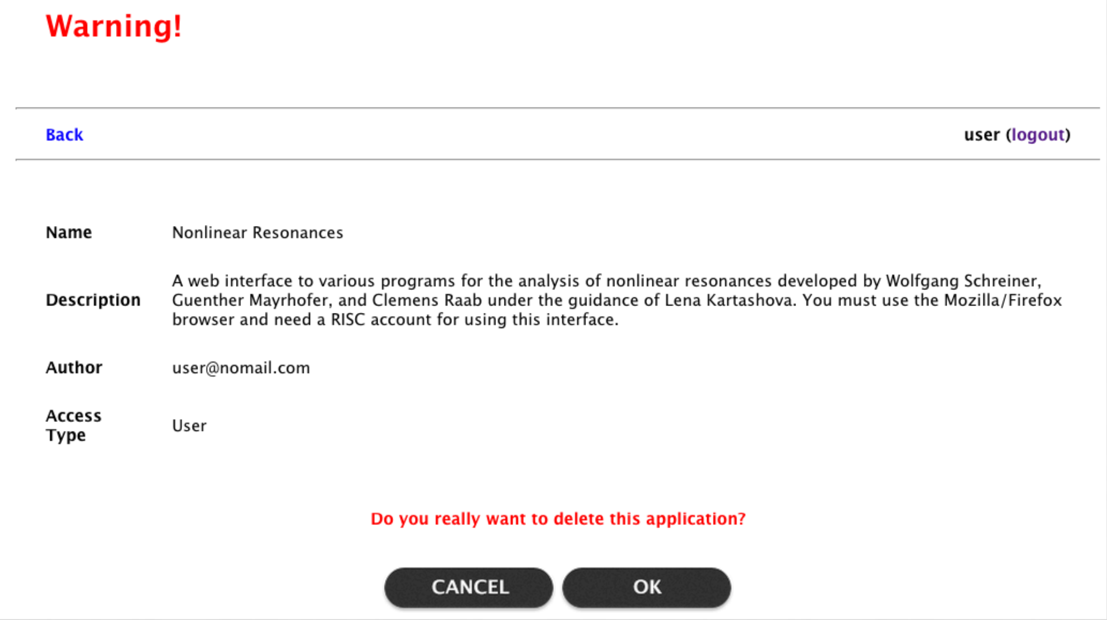
User Account: Delete Applications



All My Applications

Home   Create new application   user (logout)

Application Name	Author	Access	Created On
Nonlinear Resonances	user@nomail.com	user	2013-01-24 17:25



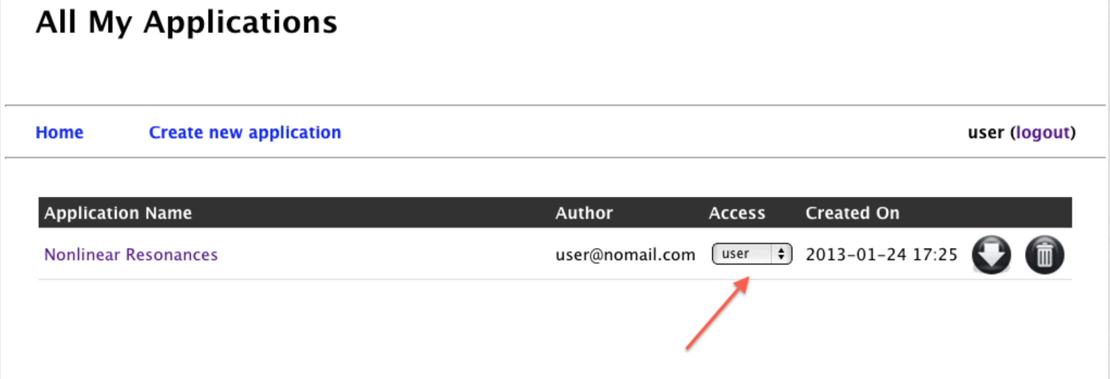
Warning!

Back   user (logout)

Name	Nonlinear Resonances
Description	A web interface to various programs for the analysis of nonlinear resonances developed by Wolfgang Schreiner, Guenther Mayrhofer, and Clemens Raab under the guidance of Lena Kartashova. You must use the Mozilla/Firefox browser and need a RISC account for using this interface.
Author	user@nomail.com
Access Type	User

Do you really want to delete this application?

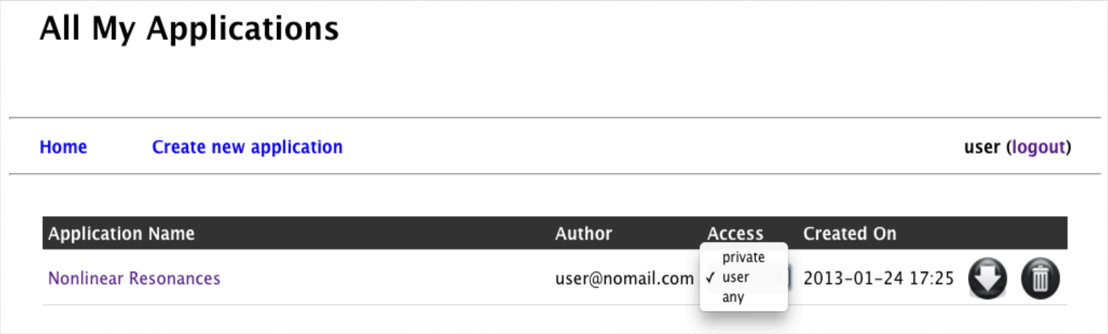
User Account: Download and Change Visibility of Applications



All My Applications

Home Create new application user (logout)

Application Name	Author	Access	Created On
Nonlinear Resonances	user@nomail.com	user	2013-01-24 17:25

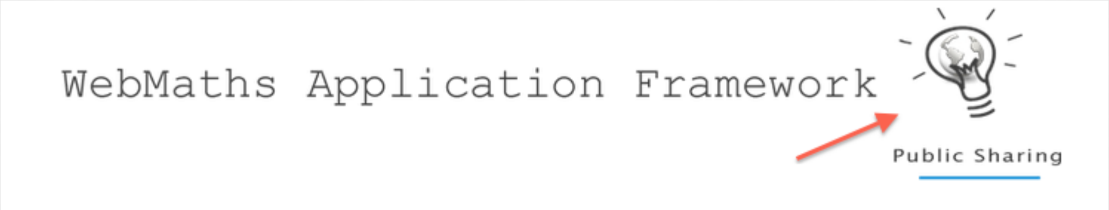


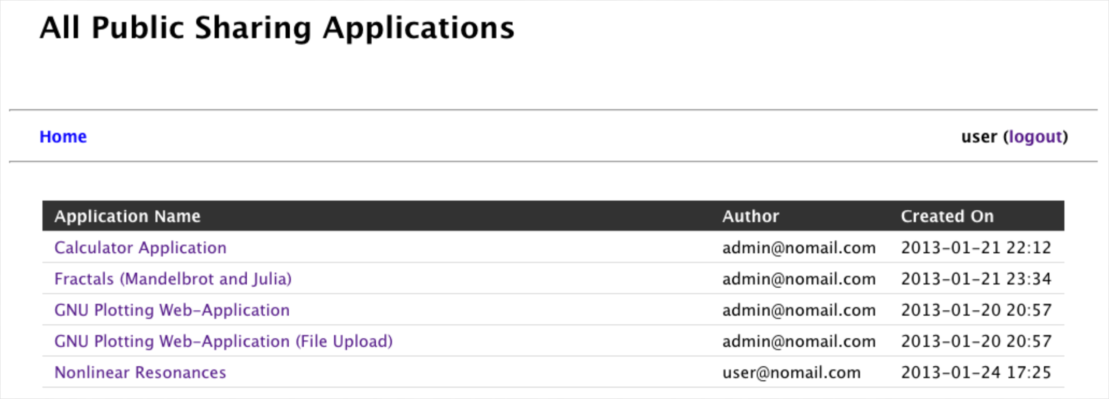
All My Applications

Home Create new application user (logout)

Application Name	Author	Access	Created On
Nonlinear Resonances	user@nomail.com	private ✓ user any	2013-01-24 17:25

User/Guest Account:
View A List of Shared Applications



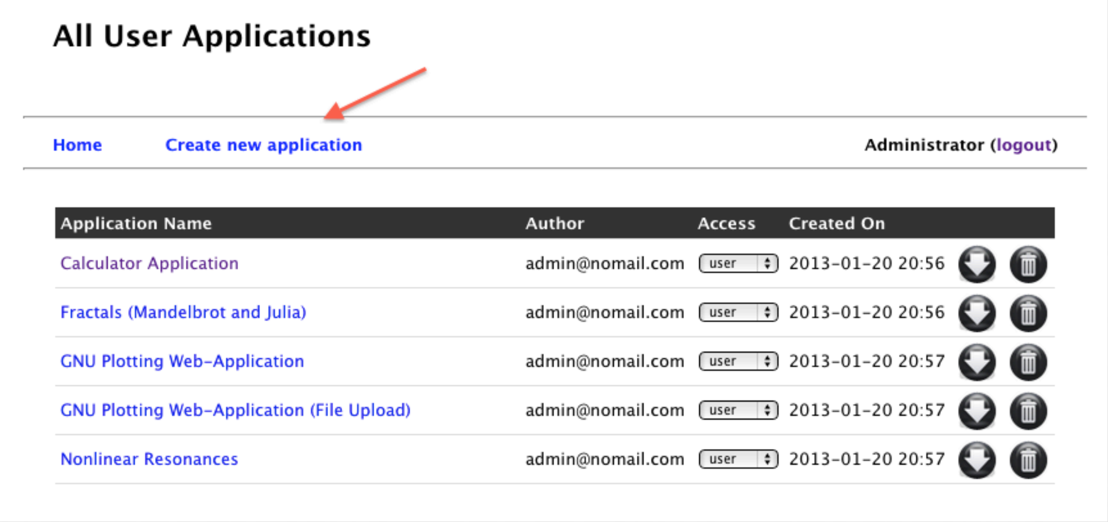


All Public Sharing Applications

Home — user (logout)

Application Name	Author	Created On
Calculator Application	admin@nomail.com	2013-01-21 22:12
Fractals (Mandelbrot and Julia)	admin@nomail.com	2013-01-21 23:34
GNU Plotting Web-Application	admin@nomail.com	2013-01-20 20:57
GNU Plotting Web-Application (File Upload)	admin@nomail.com	2013-01-20 20:57
Nonlinear Resonances	user@nomail.com	2013-01-24 17:25

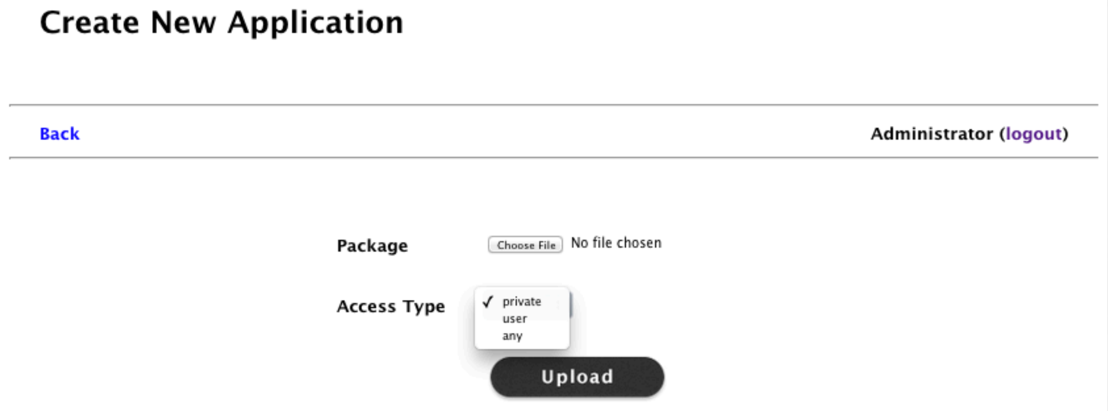
Administrator Account: Upload New Applications



All User Applications

Home   Create new application   Administrator (logout)

Application Name	Author	Access	Created On		
Calculator Application	admin@nomail.com	user	2013-01-20 20:56	⬇	🗑
Fractals (Mandelbrot and Julia)	admin@nomail.com	user	2013-01-20 20:56	⬇	🗑
GNU Plotting Web-Application	admin@nomail.com	user	2013-01-20 20:57	⬇	🗑
GNU Plotting Web-Application (File Upload)	admin@nomail.com	user	2013-01-20 20:57	⬇	🗑
Nonlinear Resonances	admin@nomail.com	user	2013-01-20 20:57	⬇	🗑



Create New Application

Back   Administrator (logout)

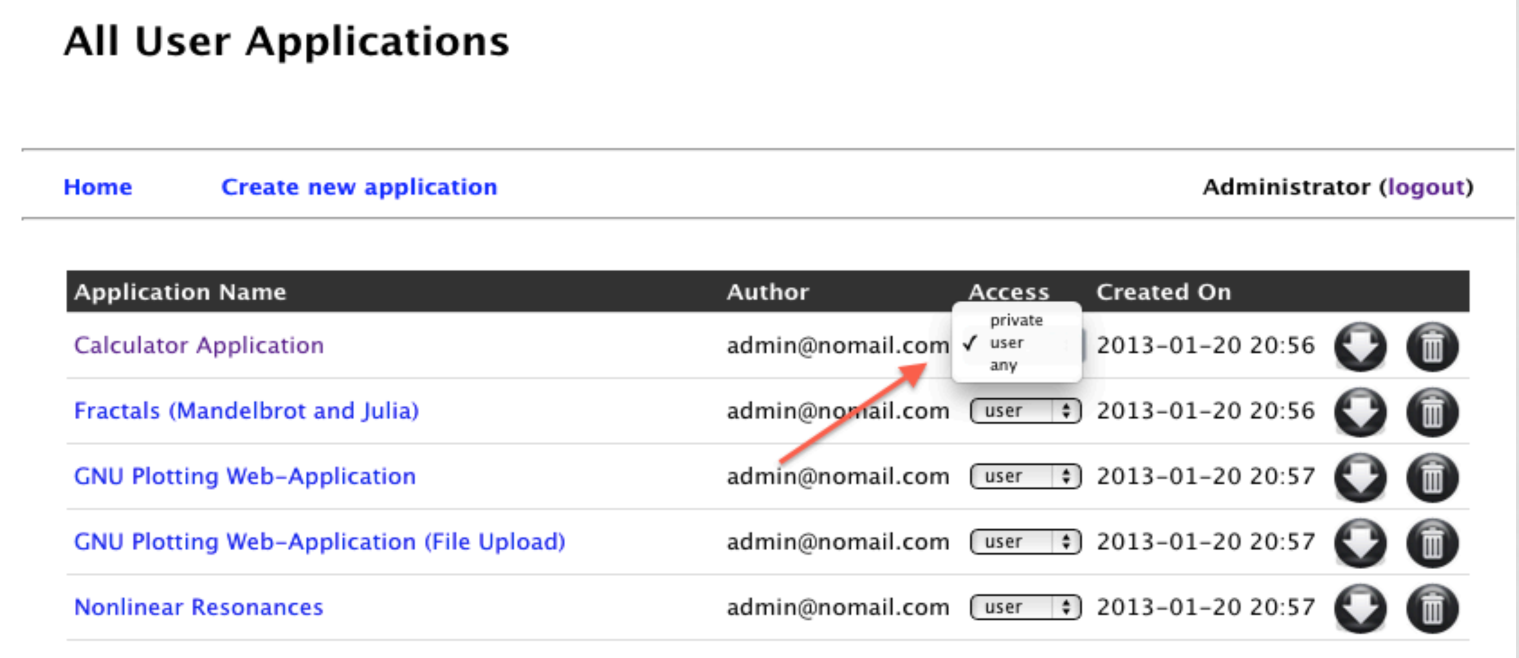
Package: Choose File No file chosen

Access Type: ✓ private / user / any

Administrator Account Application Management

WebMaths Application Framework

Public Sharing



All User Applications

Home   Create new application   Administrator (logout)

Application Name	Author	Access	Created On
Calculator Application	admin@nomail.com	private ✓ user any	2013-01-20 20:56
Fractals (Mandelbrot and Julia)	admin@nomail.com	user	2013-01-20 20:56
GNU Plotting Web-Application	admin@nomail.com	user	2013-01-20 20:57
GNU Plotting Web-Application (File Upload)	admin@nomail.com	user	2013-01-20 20:57
Nonlinear Resonances	admin@nomail.com	user	2013-01-20 20:57

Problems and Future Work

Problems:

- Pyjamas is still in an early development phase.
- Pyjamas is not fully compatible with Python language.
- Uploaded applications might post a risk to exploit the server.

Future Work:

- More security checks should be performed.
- Additional widgets and panels can be added.
- Design and define XSD for validation against XML GUI definitions.

WebMaths Framework Demo

Conclusion

- There is a need for facilities in mathematical web-based applications.
- A steep learning curve for web application development.
- Mathematician can focus solely on writing the mathematical solutions.

Thank you!