Sandia National Laboratories

College of Saint Benedict / Saint John's University

Optika

A GUI Package for Trilinos

Kurtis Nusbaum
klnusbaum@gmail.com
St. John's University
November 3, 2009

Sandia National Laboratories

College of Saint Benedict
Saint John's University

Presenation Structure

- What is Optika?
- Why might I use Optika?
- Optika features
  - Basic usage
  - Advanced features

Sandia National Laboratories

College of Saint Benedict
Saint John's University

What is Optika?

- Dynamic GUI generator
- Takes a simple set of inputs and generates a GUI
- Separation of concerns
- Examples can be found in package

Sandia National Laboratories

College of Saint Benedict
Saint John's University

Qt

- Utilized by Optika
- GUI Library from Nokia
- Cross-Platform
- Cmake integration
- http://qt.nokia.com/

Sandia National Laboratories

College of Saint Benedict
Saint John's University

Why might I use Optika?

- You want to make a canned program
- You need a user interface that isn't a complicated text file
- You don't want to spend a lot of time designing your GUI

Text File Input

Very Hard for the user to understand. Not very intuitive.

********** DIMENSION PARAMETERS **********
@ -1. -1. -1. -1. 10. Length_ref Density_ref Temp Dielec_ref VEXT_MAX
********** MESH PARAMETERS **********
@ 1 Ndim
@ 10.0 Size_x(idim): idim=0,Ndim-1
@ 0.025 Esize_x(idim): idim=0,Ndim-1
@ -1 0 Type_bc(x0,: left, right) (-1=IN_WALL, 0=IN_BULK, 1=PERIODIC, 2=REFLECT, 3=LAST_NODE)
@ 0 0 Type_bc(x1,: down, up)
@ 0 0 Type_bc(x2,: back, front)

********** FUNCTIONAL SWITCHES **********
@ 0 0 Type_func (-1=No HS functional, 0=FMT1, 1=FMT2, 2=FMT3) Type_hsdiam(0=SIGMA_DIAM 1=BH_DIAM)
@ 0 0 Type_attr Type_pairPot
(Type_attr options: -1=No attractions, 0=strict MF 2=MF_VARIABLE)
(Type_pairPot options: 0=PAIR_LJ12_6_CS, 1=PAIR_COULOMB_CS, 2=PAIR_COULOMB, 3=PAIR_YUKAWA_CS)
@ -1 Type_coul (-1=No coulomb, 0=strict MF, 1=include 2nd order corrections)
@ -1 Type_poly (-1=No polymer, 0=CMS, 1=CMS_SCFT, 2=WTC, 3=WJDC, 4=WJDC2, 5=WJDC3)

XML File

A little better, but still not quite what we want.

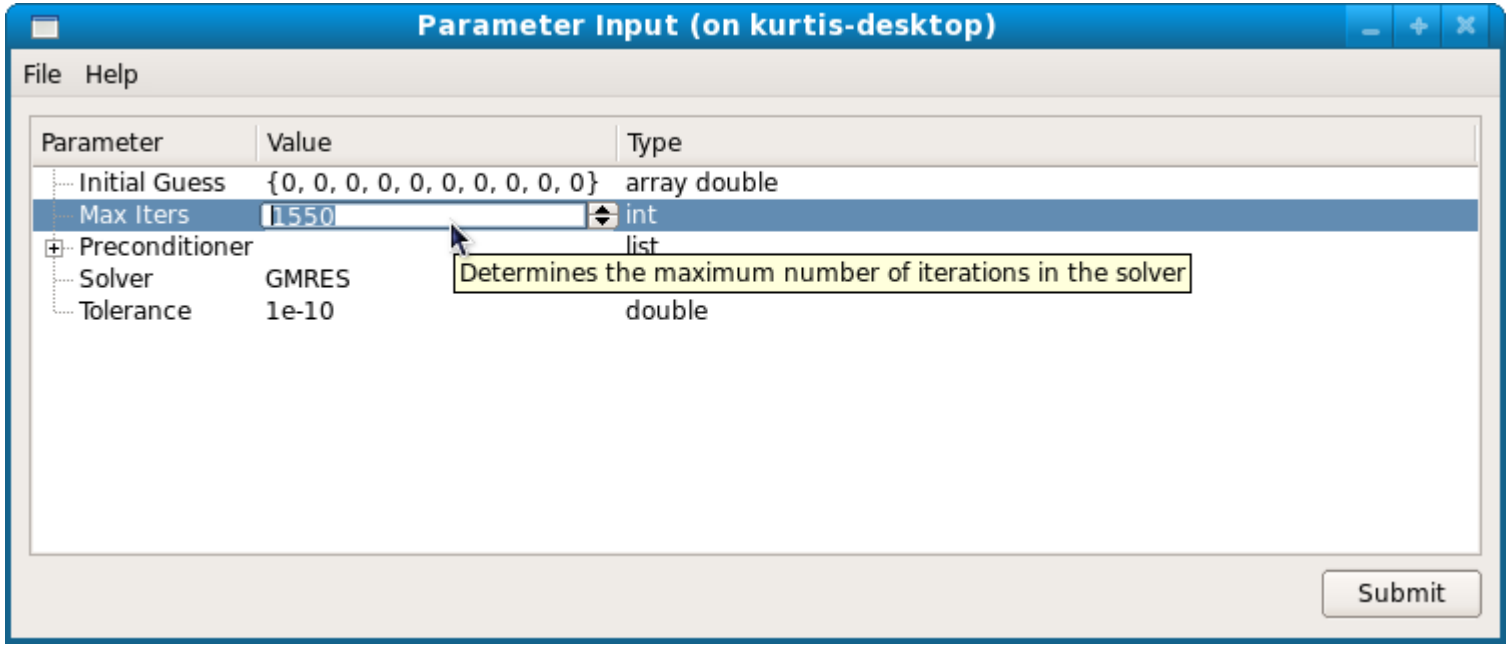
```
<?xml version="1.0" encoding="UTF-8"?>
<ParameterList>
  <Parameter name="Enable Delayed Solver Construction" value="false" type="bool"/>
  <Parameter name="Linear Solver Type" value="Amesos" type="string"/>
  <ParameterList name="Linear Solver Types">
    <ParameterList name="Amesos">
      <ParameterList name="Amesos Settings">
        <Parameter name="AddToDiag" value="0" type="double"/>
        <Parameter name="AddZeroToDiag" value="false" type="bool"/>
        <Parameter name="ComputeTrueResidual" value="false" type="bool"/>
        <Parameter name="ComputeVectorNorms" value="false" type="bool"/>
        <Parameter name="DebugLevel" value="0" type="int"/>
        <ParameterList name="Lapack">
          <Parameter name="Equilibrate" value="true" type="bool"/>
        </ParameterList>
        <Parameter name="MatrixProperty" value="general" type="string"/>
        <Parameter name="MaxProcs" value="-1" type="int"/>
```

Simple Example

```
#include "Optika_GUI.hpp"
#include "Teuchos_ParameterXMLFileReader.hpp"
#include "Teuchos_XMLParameterListWriter.hpp"

int main(int argc, char* argv[]){
	using namespace Teuchos;
	using namespace Optika;
	    //Set up Parameter List
	RCP<ParameterList> myInputs = RCP<ParameterList>(new ParameterList("Example"));
	myInputs->set("Max Iters", 1550, "Determines the maximum number of iterations in the solver");
	myInputs->set("Tolerance", 1e-10, "The tolerance used for the convergence check");
	myInputs->set("Solver" ,"GMRES" ,"The type of solver to use.");
	Teuchos::Array<double> testArray(10,0.0);
	myInputs->set("Initial Guess", testArray, "The initial guess as a RCP to an array object.");
	Teuchos::ParameterList& Prec_List = myInputs->sublist("Preconditioner",false,"Sublist that defines the
preconditioner.");
	Prec_List.set("Type", "ILU", "The type of preconditioner to use");
	Prec_List.set("Drop Tolerance", 1e-3 ,"The tolerance below which entries from the\n""factorization are
left out of the factors.");
	    //Call Optika
	getInput(myInputs);
	return 0;
}
```

Resulting GUI



Sandia National Laboratories

College of Saint Benedict / Saint John's University

Advanced Features

- Validators
  - Number validators
  - Filename validators
  - String validators
  - Array versions of most validators
- Dependencies
  - Visual Dependencies
    - Bool Visual
    - String Visual

Sandia National Laboratories

College of Saint Benedict
Saint John's University

Advanced Features Cont.

- Dependencies
  - Validator dependencies
  - Array Length dependencies
- Alternate ”exec” work-flow
- Custom Functions
  - Also available with standard work flow

Dependencies and Validators

```
Teuchos::RCP<Teuchos::StringToIntegralParameterEntryValidator<int> >
  stringFoodTypeValidator = Teuchos::rcp(
    new Teuchos::StringToIntegralParameterEntryValidator<int>(
      Teuchos::tuple<std::string>( "Cheese", "Soda", "Chips" )
      ,"Food Type"
      ));
 Teuchos::RCP<Teuchos::StringToIntegralParameterEntryValidator<int> >
  cheeseValidator = Teuchos::rcp(
    new Teuchos::StringToIntegralParameterEntryValidator<int>(
      Teuchos::tuple<std::string>( "Swiss", "American", "Super Awesome Cheese" )
      ,"Food Selector"
      ));
 Teuchos::RCP<Teuchos::StringToIntegralParameterEntryValidator<int> >
  sodaValidator = Teuchos::rcp(
    new Teuchos::StringToIntegralParameterEntryValidator<int>(
      Teuchos::tuple<std::string>( "Pepsi", "Coke", "Kurtis Cola", "Bad Cola" )
      ,"Food Selector"
      ));
 Teuchos::RCP<Teuchos::StringToIntegralParameterEntryValidator<int> >
  chipsValidator = Teuchos::rcp(
    new Teuchos::StringToIntegralParameterEntryValidator<int>(
      Teuchos::tuple<std::string>( "Lays", "Doritos", "Kurtis Super Awesome Brand" )
      ,"Food Selector"
      ));
```

Dependencies and Validators

```
Optika::StringValidatorDependency::ValueToValidatorMap testValidatorMap1;
testValidatorMap1["Cheese"] = cheeseValidator;
testValidatorMap1["Soda"] = sodaValidator;
testValidatorMap1["Chips"] = chipsValidator;


Teuchos::ParameterList& stringValiDepList = My_deplist->sublist(
                    "String Validator Dependency", false);
stringValiDepList.set("Food Selector", "Swiss", "select the food you want", cheeseValidator);
stringValiDepList.set("Food Type", "Cheese", "String Validator Dependency Tester",
        stringFoodTypeValidator);
depSheet1->addDependency(Teuchos::RCP<Optika::StringValidatorDependency>(
        new Optika::StringValidatorDependency("Food Type",
        Teuchos::sublist(My_deplist,"String Validator Dependency"),
        "Food Selector",
        Teuchos::sublist(My_deplist,"String Validator Dependency"),
        testValidatorMap1,
        cheeseValidator)));

        getInput(My_deplist, depSheet1);
        return 0;
}
```

Using Optika with Existing Code

- Optika only requires you specify the inputs you need
- Should be listed in a Teuchos Parameter List
- Any existing program with a Parameter List can use Optika
- Usually requires only minimal modification to existing source code.

Using Optika on Stratimikos

- Extract the Stratimikos functionality into a Custom Function
- Obtain inputs
- Run custom function everytime the user hits submit

Stratimikos GUI

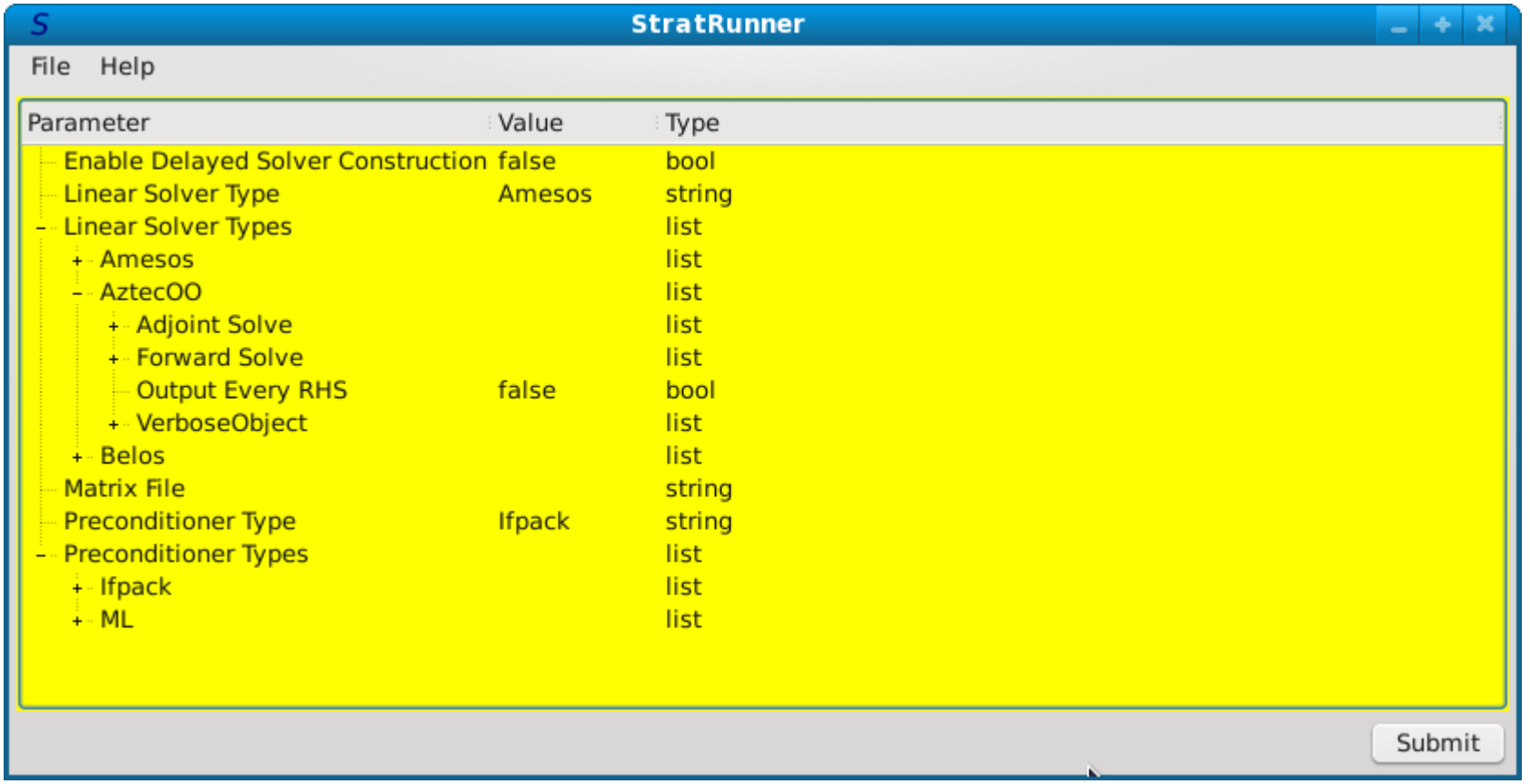
```
Int main(int argc, char* argv[]){
...
RCP<ParameterList> inputList = RCP<ParameterList>(new
        ParameterList(*linearSolverBuilder.getValidParameters()));

RCP<FileNameValidator> fileVali = ValidatorFactory::getFileNameValidator();
fileVali->setFileMustExist(true);
inputList->set("Matrix File", matrixFile, "The file containing the matrix you wish to solve", fileVali);
Optika::OptikaGUI myGUI(inputList);
myGUI.setCustomFunction(&runSolver);
myGUI.setWindowIcon("straticon.svg");
myGUI.setWindowTitle("StratRunner");
myGUI.setStyleSheet("myStyle");
myGUI.exec();
…
}
```

Sandia National Laboratories

College of Saint Benedict / Saint John's University

StratRunner



Sandia National Laboratories

College of Saint Benedict / Saint John's University

Current State Of Affairs

- Looking for user feedback
  - Optika is young
  - What do you want?
- Changes in Teuchos
  - Make Parameter List serialization better
  - Parameter List parental references
  - Move validators from Optika to Teuchos
- Add Qt as TPL
- Currently in limited release
  - Available in Trilinos repository

Sandia National Laboratories

College of Saint Benedict
Saint John's University

Summary

- Optika helps you build a better user interface
- Utilizes Teuchos Parameter Lists
- Has multiple workflows
- Has validators & dependencies
- Can be used with existing code
- Looking for user feedback
- http://trilinos.sandia.gov/packages/optika/


Sandia
National
Laboratories

Questions?


COLLEGE OF
Saint Benedict
Saint John's
UNIVERSITY