

# OpenLaszlo: A Python Success Story

PyCon 2005

Oliver Steele  
Chief Software Architect  
Laszlo Systems, Inc.



March 23, 2005

# >Lots of Languages

*Target Languages*

LZX

XHTML

*Definition Languages*

JavaScript

RELAX NG

XML

*Implementation Languages*

JavaCC

Java

Jython

Python

XSLT

# > Outline

- What we built
  - OpenLaszlo platform
  - Demos
  - Architecture
- How we built it
  - Script compiler
  - Cool features
  - Doc tools
  - Synergy
- What we learned
  - Technical challenges
  - Social challenges
  - Why it worked (and what didn't)



- Founded in 2001
- Team from Apple, Adobe, Allaire, Excite, GO, Farallon, and Macromedia
- 20 developers in Boston and San Francisco Bay
  - (Looking for 21st)
- News
  - October 5, 2004: Platform released as Open Source
  - October 13, 2004: Company receives Series B funding
  - March 2005: Earthlink announces use of Laszlo Mail



## **Develop and market Rich Internet Applications for customer-facing websites**

***Establish the Laszlo platform as the standard, open source software platform of choice for such applications***



# > Demos

- Calendar
- Dashboard



# Who Uses OpenLaszlo?



The screenshot shows the homepage of iUP.com, featuring a navigation bar with links like HOME, PLATFORMS, MAGAZINES, GENRES, SOCIALIZER, FIND GAMES, BUY GAMES, SEARCH, and LOGIN. Below the navigation is a banner for BEHR Premium Plus. The main content area includes sections for 'REVIEW' (with reviews for Star Wars Episode II and Final Fantasy VII), 'THIS YEAR' (with reviews for EA's E3 Lineup and Star Wars Episode II), and 'FAT LIST' (with reviews for EA's E3 Lineup). A sidebar on the right displays a 'BUZZOMETER' ranking of EA games.



The screenshot shows the New York Times homepage. It features a weather forecast for San Francisco, a 'My Selections' sidebar with links to Sports, News, Sports, Music, Movies, TV, and Front Page, and a 'More News' section with a 'Sports' category. The main content area displays headlines from various sections like Sports, Headlines & World News, Business, and Technology. At the bottom, there are buttons for 'Display' and 'Add More'.



The screenshot shows a color selection interface for interior design. It includes a title 'VIEW YOUR COLORS IN A ROOM', two steps: 'STEP 1 Click an Area' and 'STEP 2 Click a Color', a preview window showing a room with selected colors, and a 'Modify these colors' button. Buttons for 'BACK', 'PRINT PROJECT', and 'SAVE PROJECT' are at the bottom.



The screenshot shows the homepage of EarthLink. It features a video player with a video titled '(Re)united Bush Apologizes For Prisoner Abuse', a weather forecast for Stanford, CA (73°F, sunny), and a navigation bar with links for VIDEO, Enhanced Module, FORECAST, STORMS, PLACES, and more.



The screenshot shows a product information page for Allegra fexofenadine HCl 10 mg tablets. It includes sections for 'Pollen', 'Weather', 'Alerts/Reminders', 'Tips', 'Rebates', and 'Rate Your Allergies Today'. A chart shows pollen levels for Palo Alto over four days: Wednesday (6.4), Thursday (6.2), Friday (7.5), and Saturday (7.6). A 'SEND TO A FRIEND' button is in the top right.

**ALLERGY BUDDY**

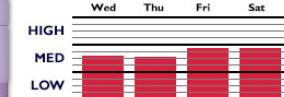
**05/12/04**

Welcome Back Susan  
Did you take your Allegra today?  
Click HERE to set reminders

**Local Pollen**   **National Pollen**   **< HOME**

**Palo Alto**

**4 DAY POLLEN FORECAST**



**ENTER ZIP CODE** 94305 **GO**

Allegra 180mg is indicated for the relief of symptoms associated with seasonal allergies in people 12 years of age and older. Side effects are low and may include headache, cold, or backache.

Nasacort AQ Nasal Spray is approved for the relief of nasal allergy symptoms caused by pet dander, dust mites and pollen. Nasacort AQ has a low incidence of side effects that may include sore throat, nosebleed, and cough. SHR-W5-1332-L

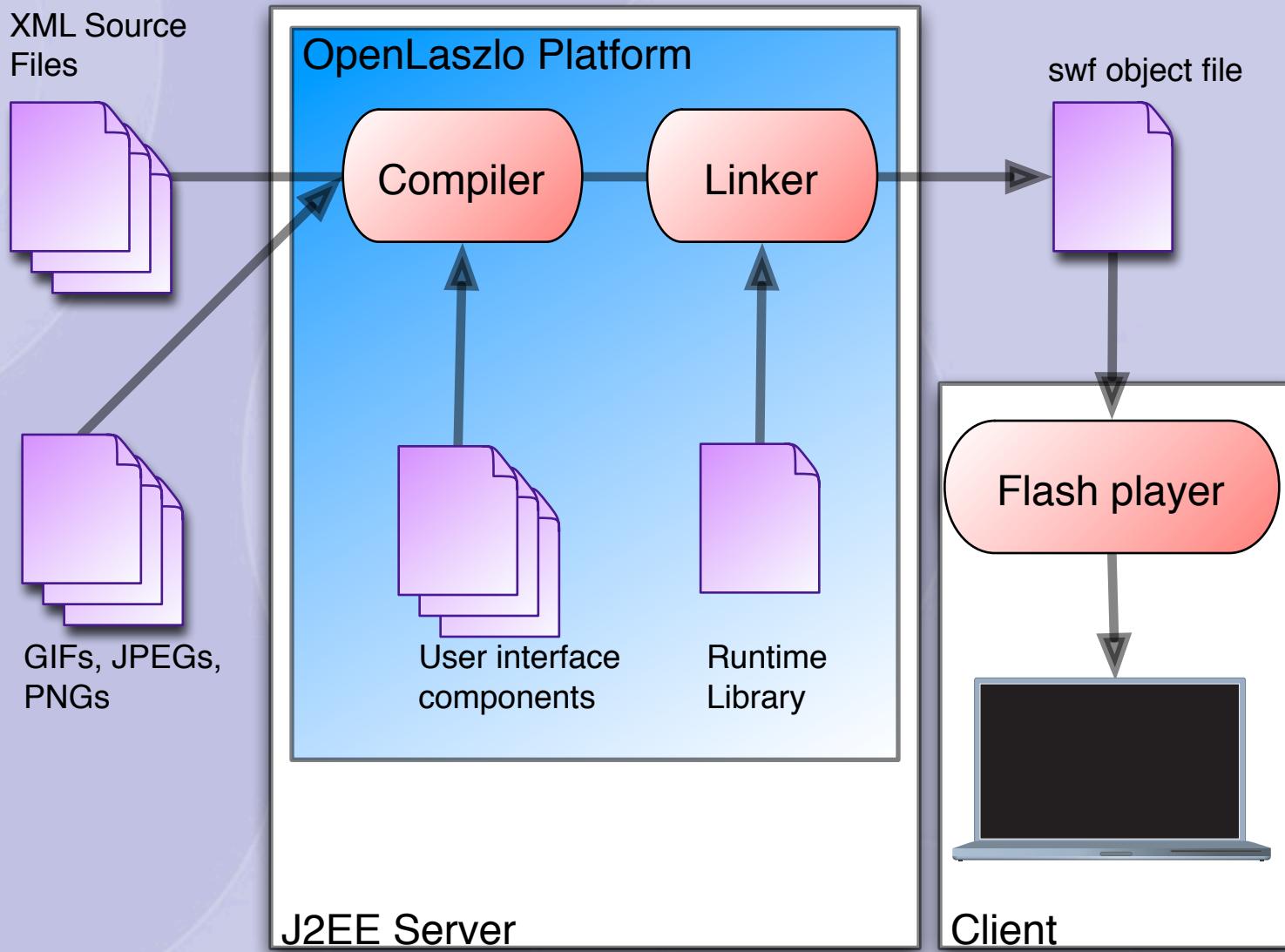


The screenshot shows a map of Texas with various locations marked. A legend indicates 'Metro area locations' (orange dot) and 'City locations' (blue cross). A sidebar provides search and reservation options. A message at the bottom asks for feedback about the service.

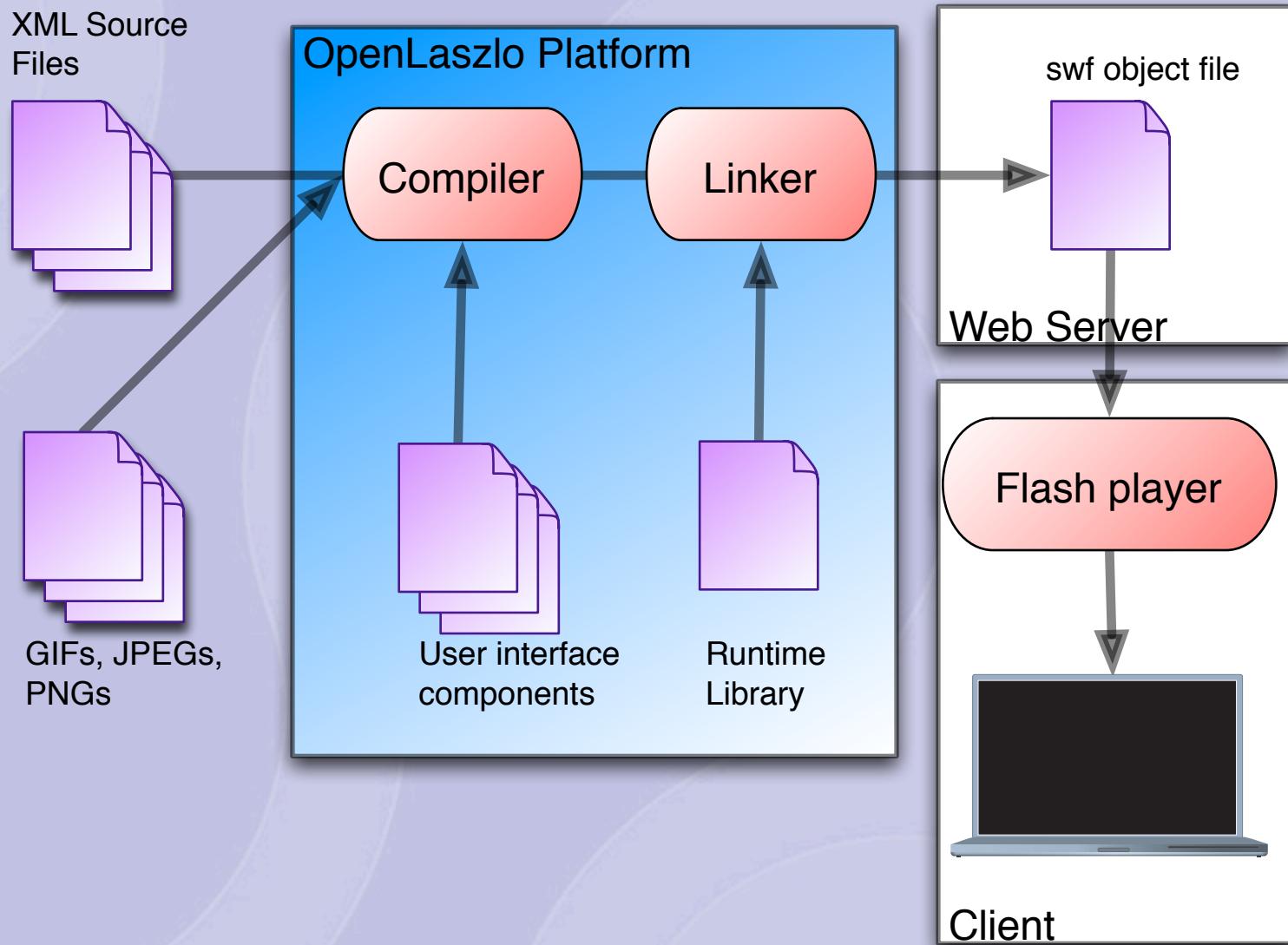


Over 20M consumers have used Laszlo-based applications

# >Under the Hood: Platform Architecture



## >Under the Hood (2): Removing the Server



## ➤ Script Compiler

- Compiles JavaScript to Flash bytecode
- Plus unanticipated cool features



## ➤ Laszlo Source Code (LZX)

```
<canvas>
  <window>
    <button
      onclick="
        animate('x', 100, 1000, true)">
      Click me!
    </button>
  </window>
</canvas>
```

- Source hierarchy mirrors object hierarchy
- Embedded JavaScript
- Constraints, data binding, and declarative states

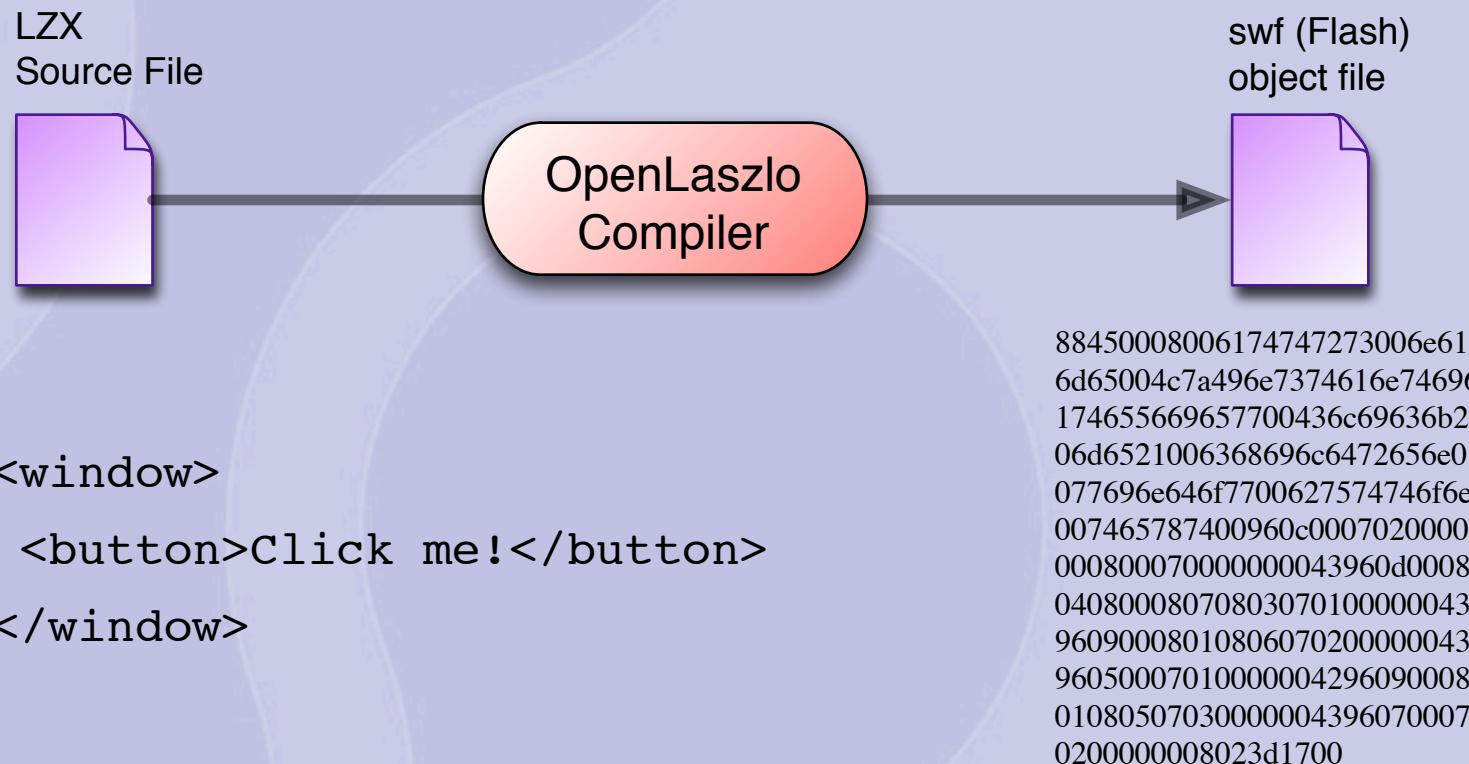


# > Script Compiler

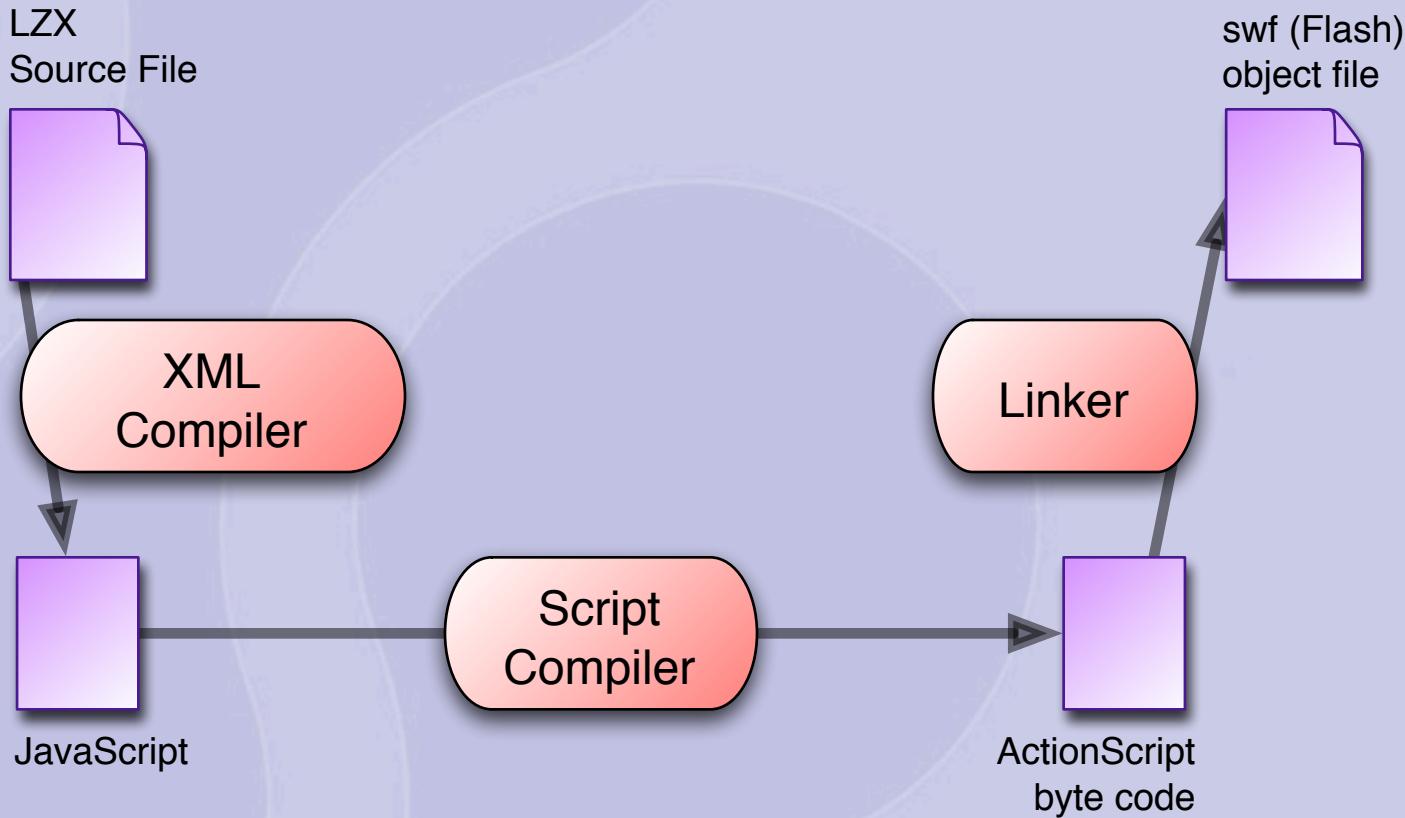
- Requirements
  - 100% Java
  - JavaScript -> ActionScript byte code
  - Three months (concurrent with language design, API design, XML compiler, and compiler architecture)
- Why not Java?
  - I'd written a lot of code in Java and Python, and didn't think I could write the Java version in time.
    - JWordNet
    - PyWordNet
- Evaluation
  - Java
  - Jython
  - SML
  - ICON



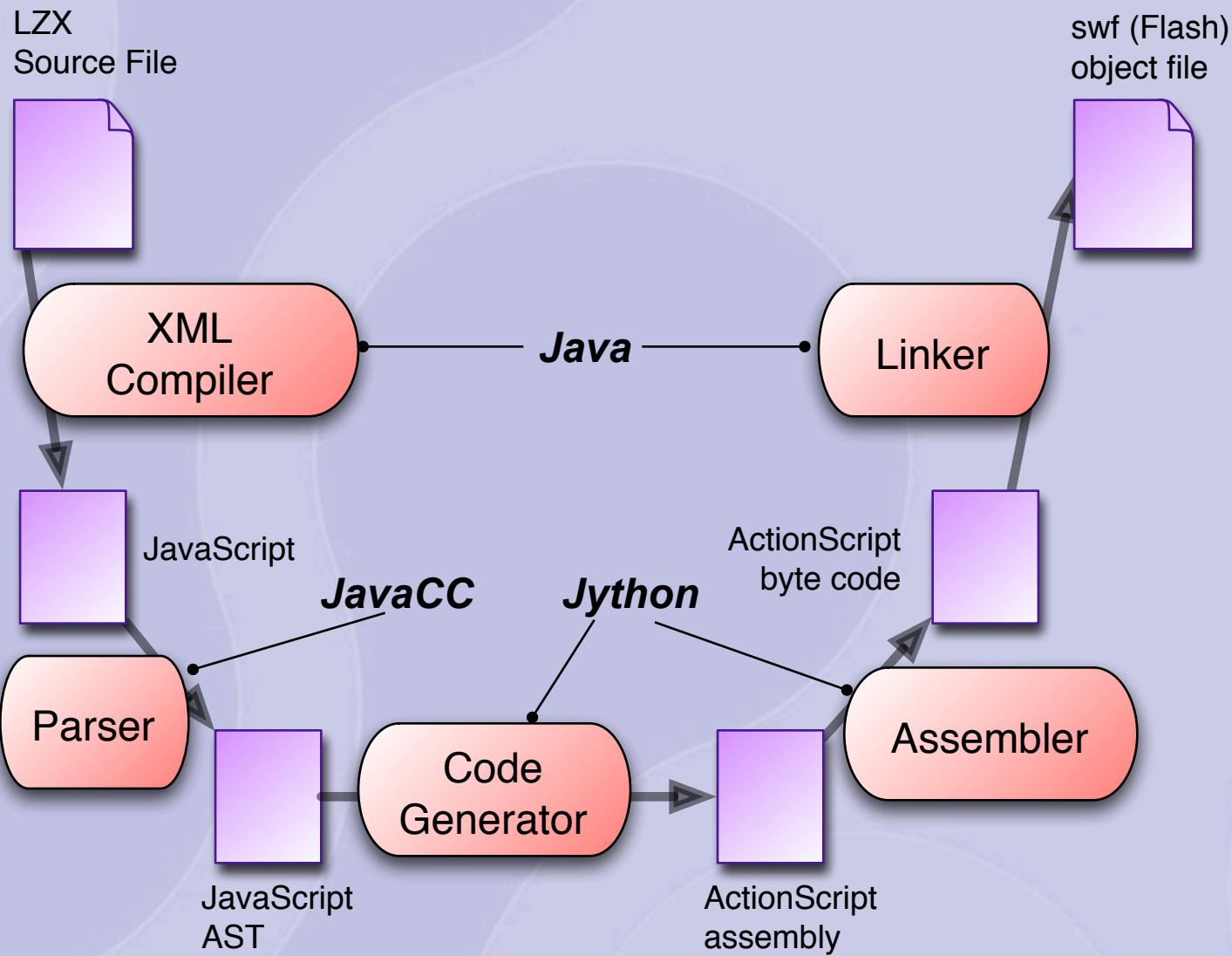
# >Script Compiler: The 10,000 foot view



# >Script Compiler: The 5,000 foot view



# >Script Compiler: The 1,000 foot view



# >Compilation Stages

## LZX (XML)

```
<window>  
  <button>Click me!</button>  
</window>
```



## JavaScript

```
LzInstantiateView(  
  {name: "window",  
   attrs: {},  
   children:  
     [{name: "button",  
      attrs: {text: "Click  
me!"}}]},  
  2);
```



## ActionScript assembly

```
constants 'attrs' 'name'  
'LzInstantiateView'  
'Click me!' 'children'  
'window' 'button' 'text'  
push '2' 'attrs' '0'  
initObject  
push 'children' 'attrs'  
'text' 'Click me!' '1'  
initObject  
push 'name' 'button' '2'  
initObject  
push '1'  
initArray  
push 'name' 'window' '3'  
initObject  
push '2'  
'LzInstantiateView'  
callFunction  
pop
```

## Byte code

88450008006
17474727300
6e616d65004c
7a496e737461
6e746961746
55669657700
436c69636b2
06d65210063
68696c64726
56e0077696e6
46f770062757
4746f6e00746
5787400960c
00070200000
00800070000
000043960d0
00804080008
07080307010
00000439609
00080108060
70200000043
96050007010
00000429609
00080108050
70300000043
96070007020
0000008023d
1700



# >Embedded JavaScript

```
<button onclick="animate('x', 100, 1000, true)"/>
```



```
LzInstantiateView(  
  {name: "button",  
   attrs: {  
     $events:  
       {onclick:  
         function $test$2Elzx_2_52_onclick_event () {  
           animate('x', 100, 1000, true)}},  
     clickable: true}}),  
  1);
```



## ► Results

- Three months
- 4.5K LOC (3K Jython, 1.5K Java + JavaCC)
- 25 bugs over three years



## > Jython Advantages

- REPL
- Literal syntax
- Doctest
- Reflection
- Java integration



```
>>> c('a=b')
constants 'a' 'b'
push 'a' 'b'
getVariable
setVariable
>>> c('a=a*2')
constants 'a'
push 'a' 'a'
getVariable
push '2'
multiply
setVariable
>>> c('function f(a,b) {return a+b}')
function2 f(r:2='a', r:1='b') ()
push 'r:2' 'r:1'
add
return
end
```



```
def substitute(self, str, **keys):
    """Parse an expression and replace any identifier with the same
    name as a keyword argument to this function, with the value of
    that key. If the value has type Splice, it's spliced into place
    instead of substituting at the same level.

>>> s = Parser().substitute
>>> s('[0,1,2]')
(ASTArrayLiteral, Literal(0.0), Literal(1), Literal(2))
>>> s('_[0,1,2]', _0=Literal("sub"))
(ASTArrayLiteral, Literal(sub), Literal(1), Literal(2))
>>> s('_[0,1,2]', _0=s('[a,b,c]'))
(ASTArrayLiteral, (ASTArrayLiteral, ASTIdentifier(a), ASTIdentifier(b),
ASTIdentifier(c)), Literal(1), Literal(2))
>>> s('_[0,1,2]', _0=Splice(s('[a,b,c]')))
(ASTArrayLiteral, ASTArrayLiteral, ASTIdentifier(a), ASTIdentifier(b),
ASTIdentifier(c), Literal(1), Literal(2))
```

N.B., there is no attempt to enforce macro hygiene  
"""



# >Concise Literal Syntax

```
DefineTests(  
    'label',  
    Stmts(['a: while (f()) b: while (g()) {break; h()}',  
           'a: while (f()) b: while (g()) {break b; h()}',  
           'a: while (f()) b: while (g()) {break a; h()}',  
           'a: while (f()) b: for (p in obj) {break; h()}',  
           'a: while (f()) b: for (p in obj) {break b; h()}',  
           'a: while (f()) b: for (p in obj) {break a; h()}',  
           'a: for (p in obj) b: while (g()) {break; h()}',  
           'a: for (p in obj) b: while (g()) {break b; h()}',  
           'a: for (p in obj) b: while (g()) {break a; h()}',  
           'a: for (p in obj) b: for (p in obj) {break; h()}',  
           'a: for (p in obj) b: for (p in obj) {break b; h()}',  
           'a: for (p in obj) b: for (p in obj) {break a; h()}',  
    ]))
```



# ➤ Reflection

```
def visit(self, node):
    fn = getattr(self, 'visit' + node.name)
    fn(node, *node.children)

def visitArrayLiteral(self, node, *args):
    ...

def visitBinaryExpressionSequence(self, node, a, op, *args):
    ...

def visitFunctionDeclaration(self, node, *args):
    ...

def visitIdentifier(self, node):
    ...

def visitLiteral(self, node):
    ...
```



## ➤ Jython disadvantages

- Execution speed
- Deployment issues



## ➤ Port Assembler to Java

- 1.2KLoC Jython -> 2.7KLoC Java
- Harder to work with
- But...10 times as fast



## ➤ Deployment Issues

- Deploying Jython within a J2EE Servlet has (had?) severe class loader issues
- Jython triggered a memory manager error in JVM 1.4
- Jython 2.0/Java 1.5 issues



## ➤ However...

- Xalan, Xerces, Jing, Commons Logging, and JDOM all had issues too



## ➤ The First Pattern

- Jython saved us a lot of time during development, but cost (less) time during deployment.
- But so did several other Java libraries
- But this still makes it difficult to justify, especially to deployers (who don't see the development-time advantages)



➤ And then we were done...



# >Constraint System (Laszlo 1.0)

## Requirements

- Width of view is a *function of parent's width and view's 'border' property*
- Width is updated whenever parent's width changes
- Width is updated whenever 'border' property value changes

## Observer Pattern

```
<view>
  <method event="oninit">
    registerListener(parent, 'width', myWidthListener);
    registerListener(this, 'border', myWidthListener);
    myWidthListener();
  </method>
  <method name="myWidthListener">
    this.width = parent.width - 2*this.border;
  </method>
</view>
```

## Constraint expression

```
<view width="${parent.width - 2*this.border}" />
```



# >Constraint Implementation

```
<view width="${parent.width - 2*this.border}" />
```



```
LzInstantiateView(  
  {  
    attrs: {  
      name: "view",  
      $refs: {  
        width: function $test$2Elzx_2_46_width_always () {  
          this.setAttribute("width",  
            parent.width - 2*this.border)}}}},  
  1);
```



```
$test$2Elzx_2_46_width_always.dependencies =  
  [parent, 'width', this, 'border']
```



➤ And then we were done...

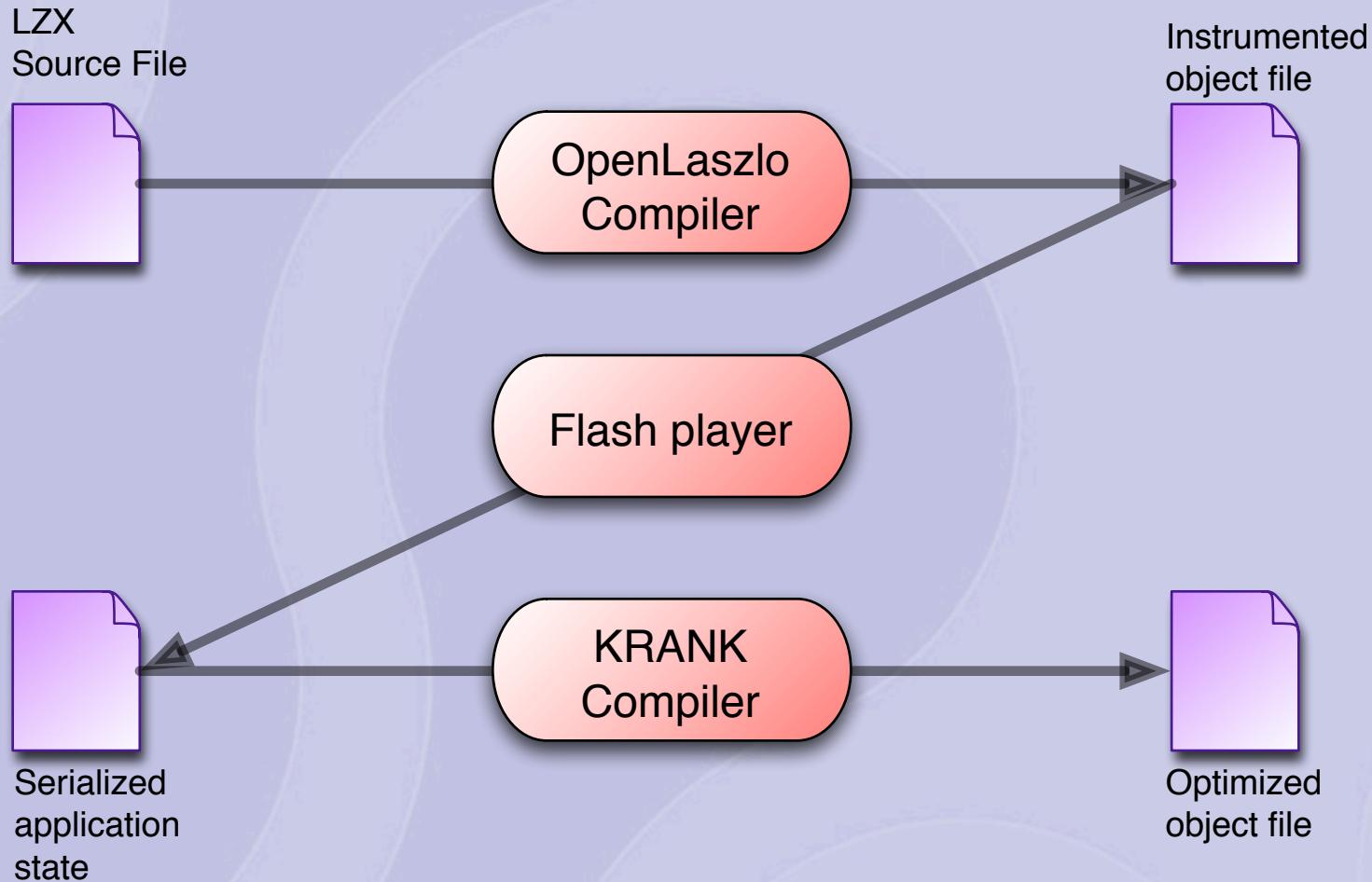


## ➤ KRANK Feature (Laszlo 2.0)

- Problem: Slow initialization time
- Solution: Initialize the application prior to deployment
- Results: 2-4x performance increase, at expense of developer time and application size



# >KRANK feature



➤ But now we're really done..



## ➤ The Second Pattern

- Python (Jython) is good for prototyping
- But, you don't know when you're done prototyping



Contents Classes Tags

Find

Welcome

- ▶ Structure
- ▶ View Basics
- ▼ Components
  - alert
  - button
  - checkbox
  - combobox
  - datepicker
  - edittext
  - floatinglist
  - form
  - grid
  - gridcolumn
  - gridtext
  - hscrollbar
  - list
  - listitem
  - menu
  - menubar
  - menuitem
  - menuseparator
  - modaldialog
  - radiobutton
  - radiogroup
  - scrollbar

If `horizontalScrollbar` is `false`, the grid will never show a horizontal scrollbar, even if the rows are wider than the grid.

### Attributes inherited from `Basecomponent`

`doesenter, enabled, hasdefault, isdefault, style, styleable, text`

### Attributes inherited from `Basegrid`

`bcolor0, bcolor1, columns, contentdatapath, hilite, multiselect, rowheight, selectable, showhlines, shownitems, showvlines, sizetoheader, spacing`

### Attributes inherited from `Node`

`classroot, cloneManager, datapath, id, ignoreAttribute, immediateparent, initstage, name, nodeLevel, onconstruct, oninit, parent, placement, subnodes`

### Attributes inherited from `View`

`align, bacolor, clickable, clickregion, clip, cursor, defaultplacement, fcolor, focusable, focustrap, font, fontsize, fontstyle, frame, framesloadratio, hassetheight, hassetwidth, height, layout, loadratio, mask, onblur, onclick, ondata, ondblclick, onfocus, onkeydown, onkeyup, onmousedown, onmouseout, onmouseover, onmouseup, onselect, opacity, options, pixellock, play, resource, resourceheight, resourcewidth, rotation, selected, selectiontype, source, stretches, subviews, totalframes, unstretchedheight, unstretchedwidth, valign, visible, width, x, xoffset, xscale, y, yoffset, yscale`

### Methods

#### Methods inherited from `basecomponent`

`doEnterDown, doEnterUp, doSpaceDown, doSpaceUp, setStyle, setTint, updateDefault`

#### Methods inherited from `basegrid`

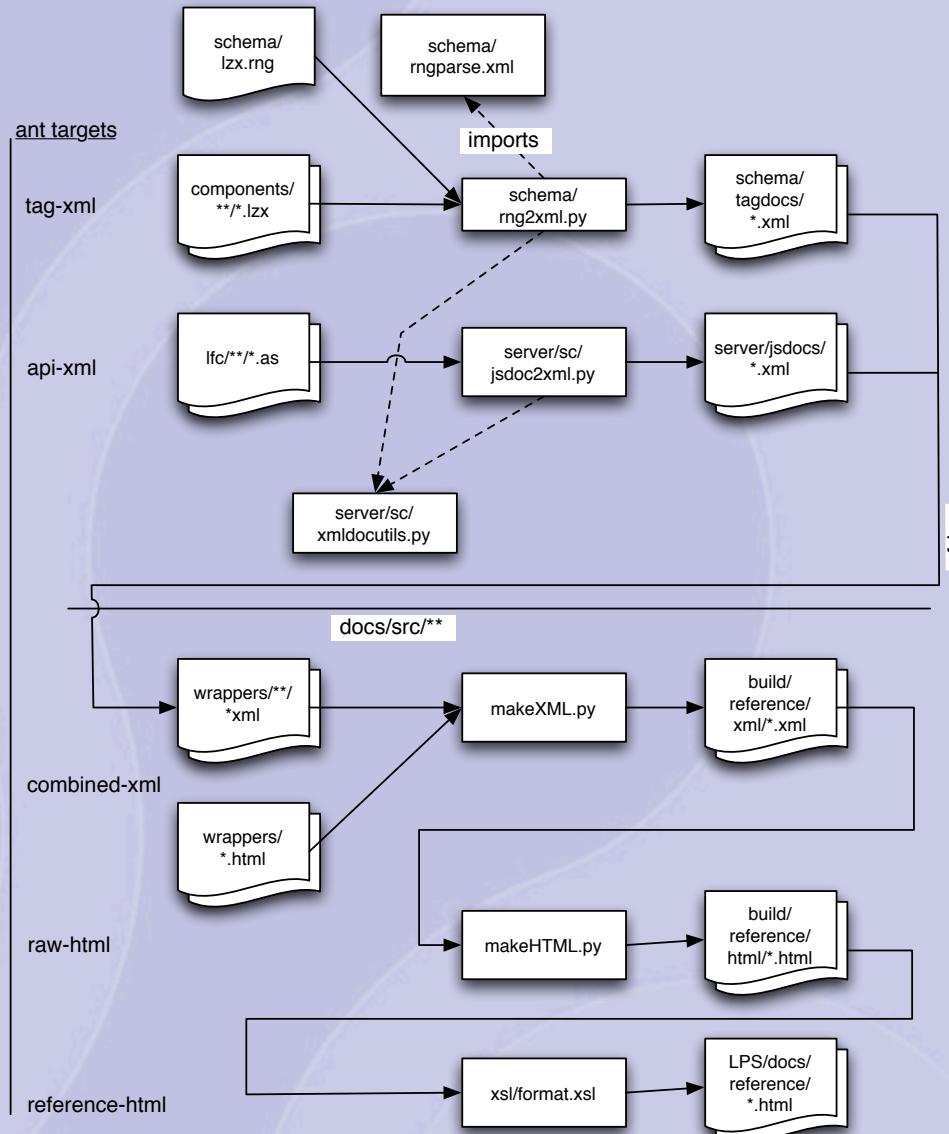
`clearSelection, clearSort, getIndexForItem, getItem, getItemAt, getNumItems, getSelection, removeItemAt, select, selectItem, selectItemAt, selectNext, selectPrev`

#### Methods inherited from `IzNode`

`animate, applyConstraint, applyData, childOf, completeInstantiation, createChildren, dataBindAttribute, destroy, determinePlacement,`



# OpenLaszlo Doc Tools



# >Lots of Languages: Developers Guide

*Target Languages*

LZX

XHTML

*Definition Languages*

JavaScript

RELAX NG

XML

*Implementation Languages*

JavaCC

Java

Jython

Python

XSLT



# >Lots of Languages: LZX Reference

*Target Languages*

LZX

XHTML

*Definition Languages*

JavaScript

RELAX NG

XML

*Implementation Languages*

JavaCC

Java

Jython

Python

XSLT

Utility Library



## > The Third Pattern

- There are different requirements for build system and deployment
- Components may move between build system and deployment...
- ...or be shared by both



# >Conclusions



## > Experimentation

- Let us experiment with new features:
  - Krank
  - Constraints
  - PUSH merge
  - Macros
- Swings of “I wish we had ported” vs. “I’m glad we didn’t port yet”
- Don’t know when you’re done prototyping

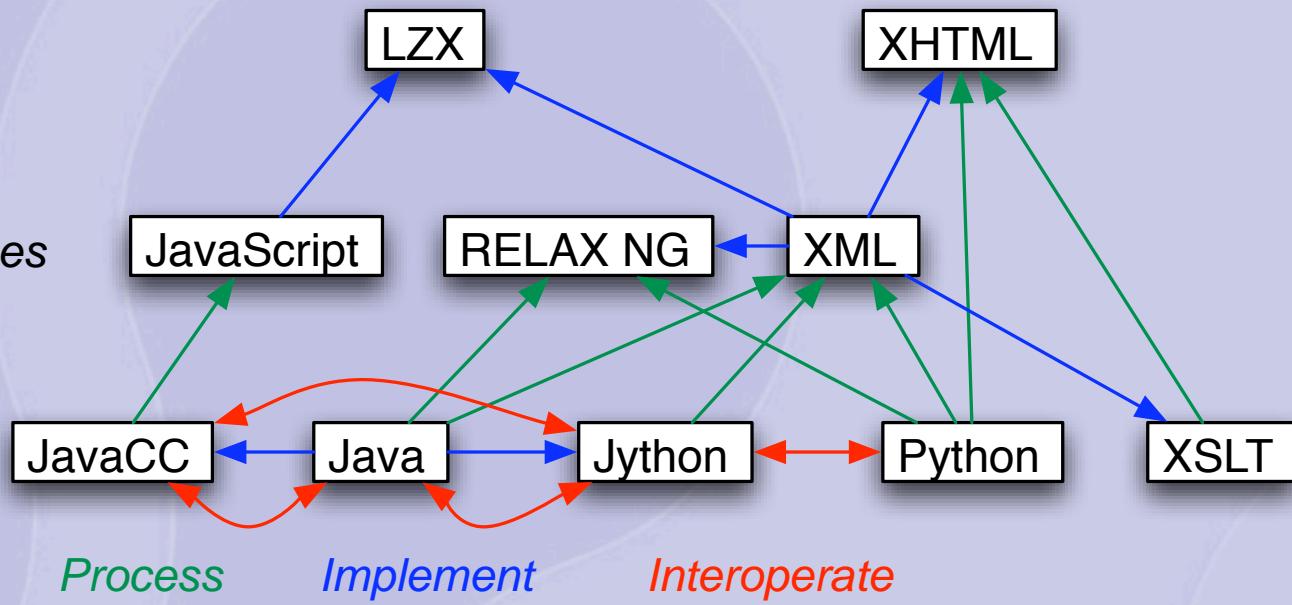


# >Lots of Languages (2)

*Target Languages*

*Definition Languages*

*Implementation Languages*



## > Migration

- Smooth path between Python and Jython
- Possible path between Jython and Java
- Excellent integration between Jython and Java-based tools



## ➤ What would help Jython?

- Smoother path to performance
- Smoother path to deployment



