XForms: An Interactive Forms Generator

Masters Candidate:	**Nolan W. Whitaker**
Faculty Advisor:	**Dr. Raphael Finkel**
Committee Members:	**Dr. Jaromczyk** **Dr. Seales**

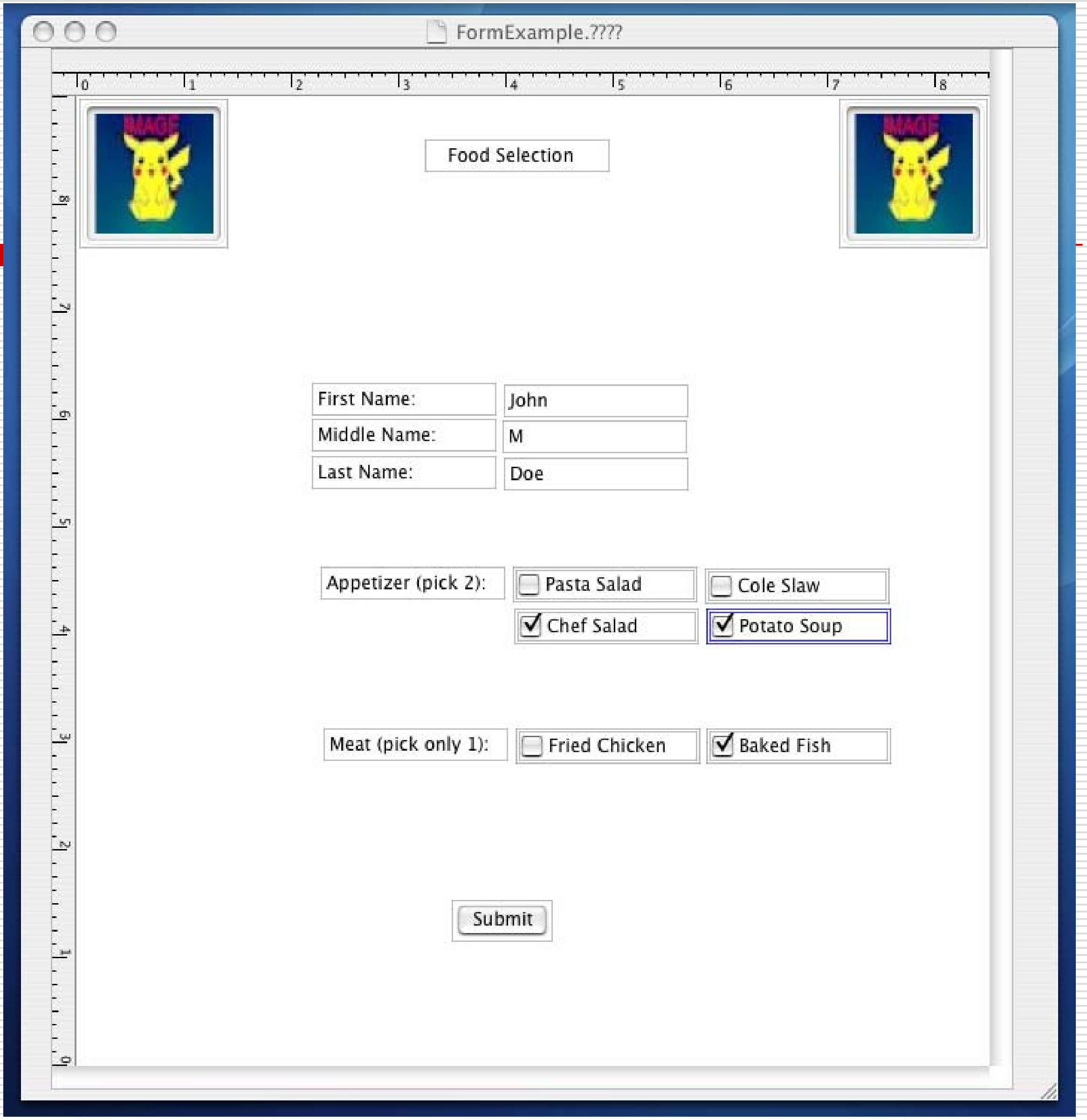
Overview

- XForms is a versatile, Mac OS X forms design tool that automatically generates forms in multiple media formats from a single design source. XForms can generate traditional paper-based forms and/or a collection of web pages with code to collect and store all the data on the form.

- XForms is an application written in Objective-C using Apple’s Cocoa framework as a proof of concept application designed to be easily extendable to support any number of form output formats.

XForms

Page Design View


FormExample.????

Food Selection

First Name:	John
Middle Name:	M
Last Name:	Doe

Appetizer (pick 2):	☐ Pasta Salad	☐ Cole Slaw
	✓ Chef Salad	✓ Potato Soup
Meat (pick only 1):	☐ Fried Chicken	✓ Baked Fish

XForms

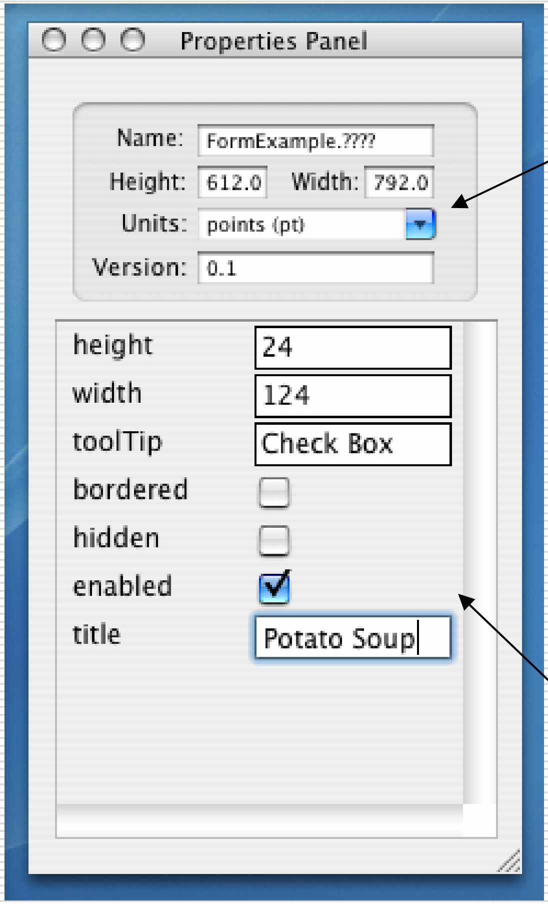
Attributes for each form page were designed in Interface Builder at compile time.

Properties Panel View

Attributes for each form element are generated dynamically at run time as the element is selected from the form page view.

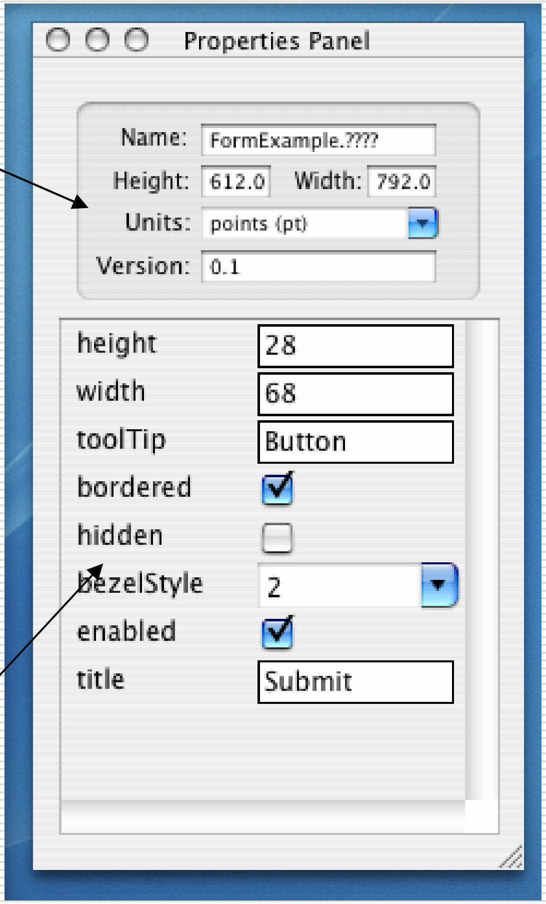
Static Design

Dynamic Design



Name:	FormExample.????
Height:	612.0
Width:	792.0
Units:	points (pt)
Version:	0.1

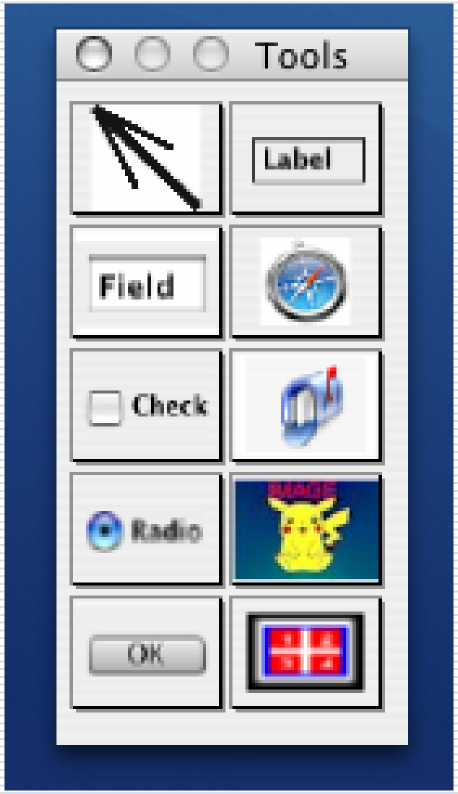
height	24
width	124
toolTip	Check Box
bordered	☐
hidden	☐
enabled	✓
title	Potato Soup



Name:	FormExample.????
Height:	612.0
Width:	792.0
Units:	points (pt)
Version:	0.1

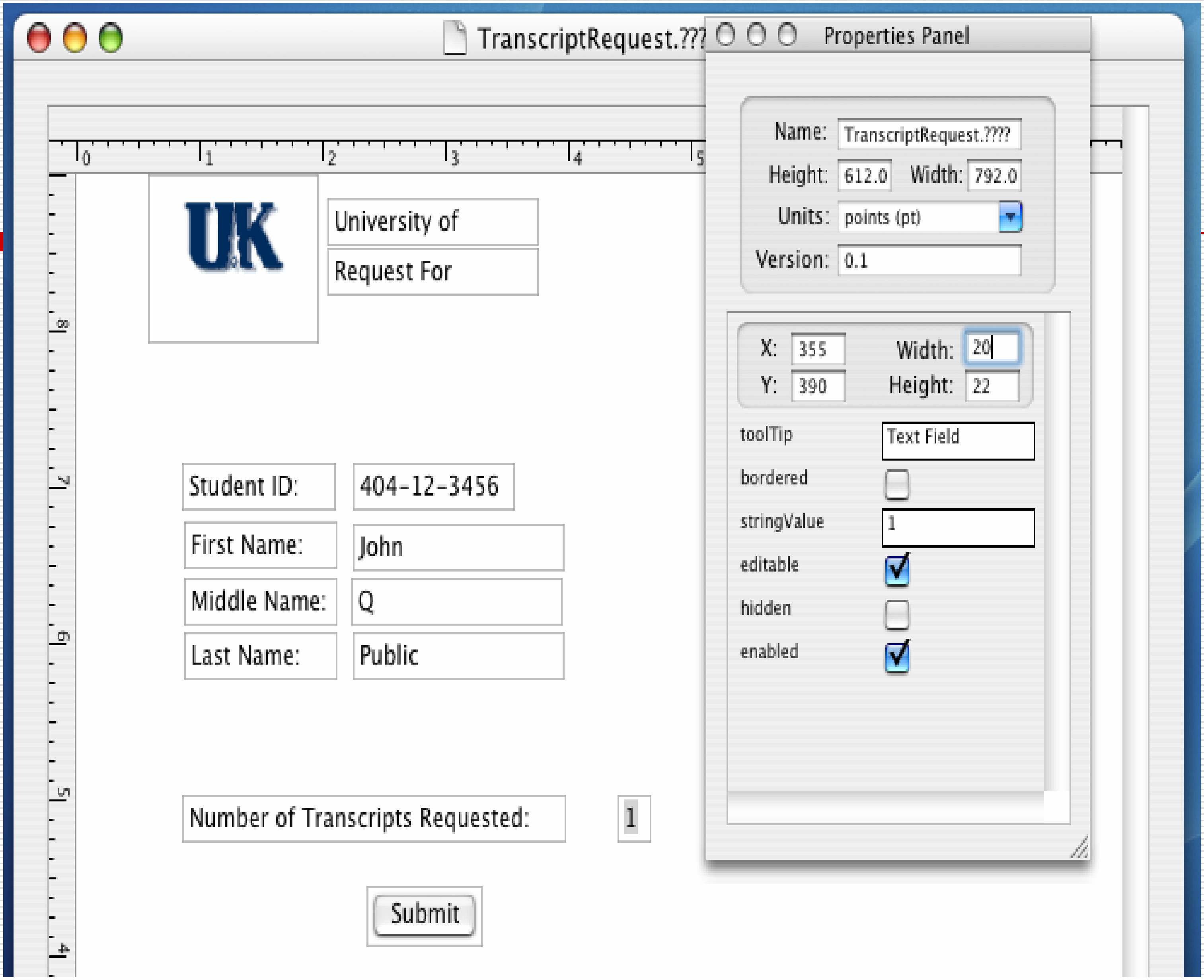
height	28
width	68
toolTip	Button
bordered	✓
hidden	☐
bezelStyle	2
enabled	✓
title	Submit

XForms



Tool Pallet View

- Dynamically generated from a dictionary
- Drag and Drop Functionality


TranscriptRequest.???

University of

Request For

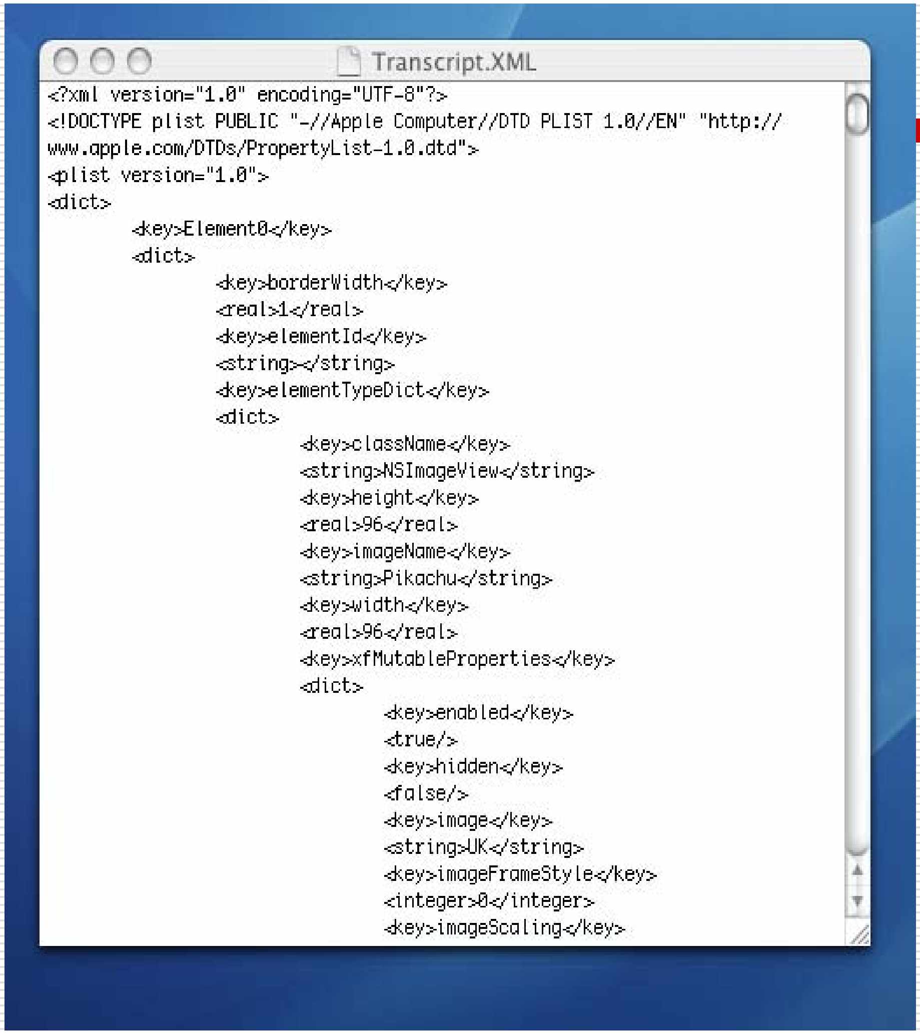
Student ID:	404-12-3456
First Name:	John
Middle Name:	Q
Last Name:	Public
Number of Transcripts Requested:	1

Properties Panel

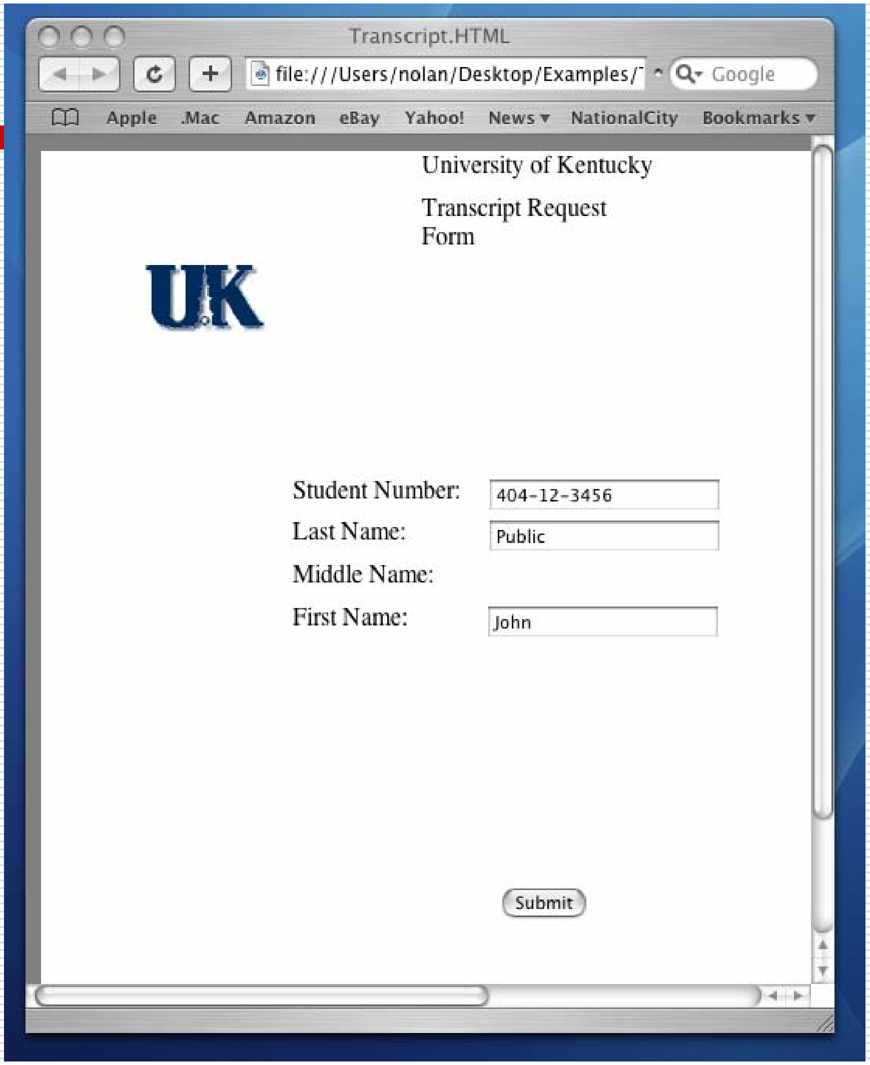
Name: TranscriptRequest.????
Height: 612.0 Width: 792.0
Units: points (pt)
Version: 0.1

X: 355 Width: 20
Y: 390 Height: 22

toolTip	Text Field
bordered	☐
stringValue	1
editable	✓
hidden	☐
enabled	✓



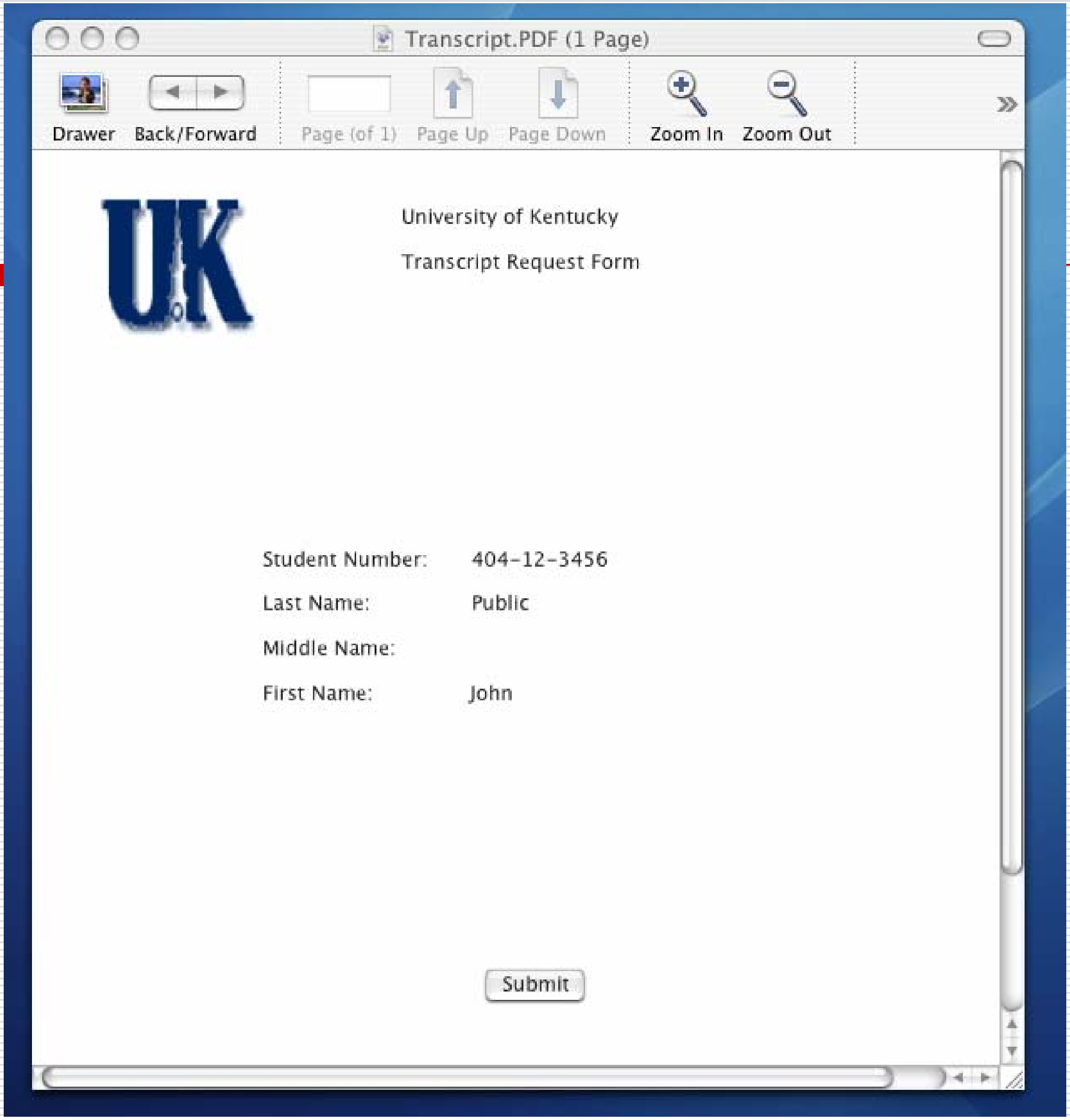
```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple Computer//DTD PLIST 1.0//EN" "http://
www.apple.com/DTDs/PropertyList-1.0.dtd">
<plist version="1.0">
<dict>
        <key>Element0</key>
        <dict>
                <key>borderWidth</key>
                <real>1</real>
                <key>elementId</key>
                <string></string>
                <key>elementTypeDict</key>
                <dict>
                        <key>className</key>
                        <string>NSImageView</string>
                        <key>height</key>
                        <real>96</real>
                        <key>imageName</key>
                        <string>Pikachu</string>
                        <key>width</key>
                        <real>96</real>
                        <key>xfMutableProperties</key>
                        <dict>
                                <key>enabled</key>
                                <true/>
                                <key>hidden</key>
                                <false/>
                                <key>image</key>
                                <string>UK</string>
                                <key>imageFrameStyle</key>
                                <integer>0</integer>
                                <key>imageScaling</key>
```



University of Kentucky

Transcript Request Form

Student Number:	404-12-3456
Last Name:	Public
Middle Name:	
First Name:	John



University of Kentucky

Transcript Request Form

Student Number:	404-12-3456
Last Name:	Public
Middle Name:	
First Name:	John

Project Goals

- Learn the Objective-C language
- Learn Apple Computer’s Cocoa Software Development Framework
- Implement a software project using the Model-View-Controller design pattern paradigm without mixing model, view, or controller classes
- Explore topics related to visual code generation and design

Objective-C is a object-oriented extension to the C programming language

- Key Features
  - Dynamism
    - Dynamic Typing, Binding, Loading (self modifying code)
  - Messaging
    - Exceptions, Nil-Targeted Messages (Responder Chains)
  - Memory Management
    - Reference Counts, Auto-release Pools

Performance Against Goals

- Goal: Learn the Objective-C language
  - Objective-C is a object-oriented extension to the C programming language.
- Actual: SUCCESS learning usage of
  - Memory Management using Reference Counts
    - retain / release
  - Object Messaging and Key-Value Class Encoding
    - [foo setValue:fooVal forKey:keyName];

Performance Against Goals

- Goal: Learn the Cocoa Framework
  - Apple’s official Mac OS X software development framework based on NeXTStep/OpenStep
- Actual: SUCCESS
  - Gained framework experience:
    - Constructing Key-View Coding classes
    - Constructing custom Views and controls (widgets) programmatically and via the GUI InterfaceBuilder
    - Intra-application communication via notifications and direct messaging

Performance Against Goals

- Goal: Learn the Cocoa Framework (cont)
- Actual: SUCCESS
  - Cocoa is a complex and evolving development framework with few references available.
  - Gained framework experience:
    - Using delegates instead of sub-classing objects
      - Delegates are helper objects for a class instance
      - Delegates allow a developer to modify the behavior of an object without sub-classing.
      - Delegates are sent messages when to the parent object’s status changes and queried for information as needed

Performance Against Goals

- Goal: Learn the Cocoa Framework (cont)
- Actual: SUCCESS
  - Gained framework experience:
    - Using Cocoa memory management Autorelease Pools
    - Using Cocoa’s NSDocument architecture for document-based applications
    - Using Cocoa’s abstract data types
      - Instances of NSDictionary and NSArray are the primary internal container data structures for XForm documents

Performance Against Goals

- Goal: Model-View-Controller Design Pattern
  - **Design patterns** are “simple and elegant solutions to specific problems in object oriented software design.”
  - MVC is a collection three or more classes separating the application’s data model, user interface, and control structure.
  - Supports code reuse
    - By decoupling the user interface and data model from an application’s control structures, proponents of MVC argue that both the data model and GUI classes become better candidates for re-use as is or with little modification

Performance Against Goals

- Actual: SUCCESS implemented a pure MVC design without mixing model, view, or controller classes
  - Model Classes
    - XFForm, XFElement
  - View Classes
    - XFPageView, XFElementView, XFToolView
  - Controller Classes
    - XFDocument, XFPropertyController

Performance Against Goals

- Goal: Explore topics related to visual code generation and design
- Actual: SUCCESS
  - User Experience
    - User Interface and Presentation Issues
    - Custom Control and Tool Pallet Design
    - Page Layout & Pagination
      - Mixed Media Issues: Paper vs Web vs Computer Display
  - Application Interoperability
    - File Formats
      - XML vs Proprietary Formats
    - Target Platforms
      - Supporting multiple platforms with mixed feature support

XForms

- Related work (Closed & open source)
  - Mixed Media Design Tools
    - **Adobe InDesign CS2**
    - Microsoft Office
      - Word and PowerPoint were can generate web pages from a native document but were not created to be mixed media design tools
  - Single Media Form Design Tools

Paper:	JetForm, FormFlow, Adobe Acrobat
Application:	Glade, InterfaceBuilder, Visual Studio
Web:	Macromedia Studio, Visual InterDev

Key Lessons For Cocoa Developers

- **Use GUI design tools like InterfaceBuilder**
  - Designing an application interface programmatically is very time-consuming and tedious– don’t unless you must.
- **Use Delegates When Possible**
  - Use the composition design pattern to create a class composed of framework class instances and delegate methods rather than subclass.
- **Let the framework do the work for you**
  - Frameworks can be designed in a way that doesn’t significantly impact performance and yet provides incredible funcationality. Find a good one and use it.

Questions & Comments

QUESTIONS?

Comments?

Concerns?