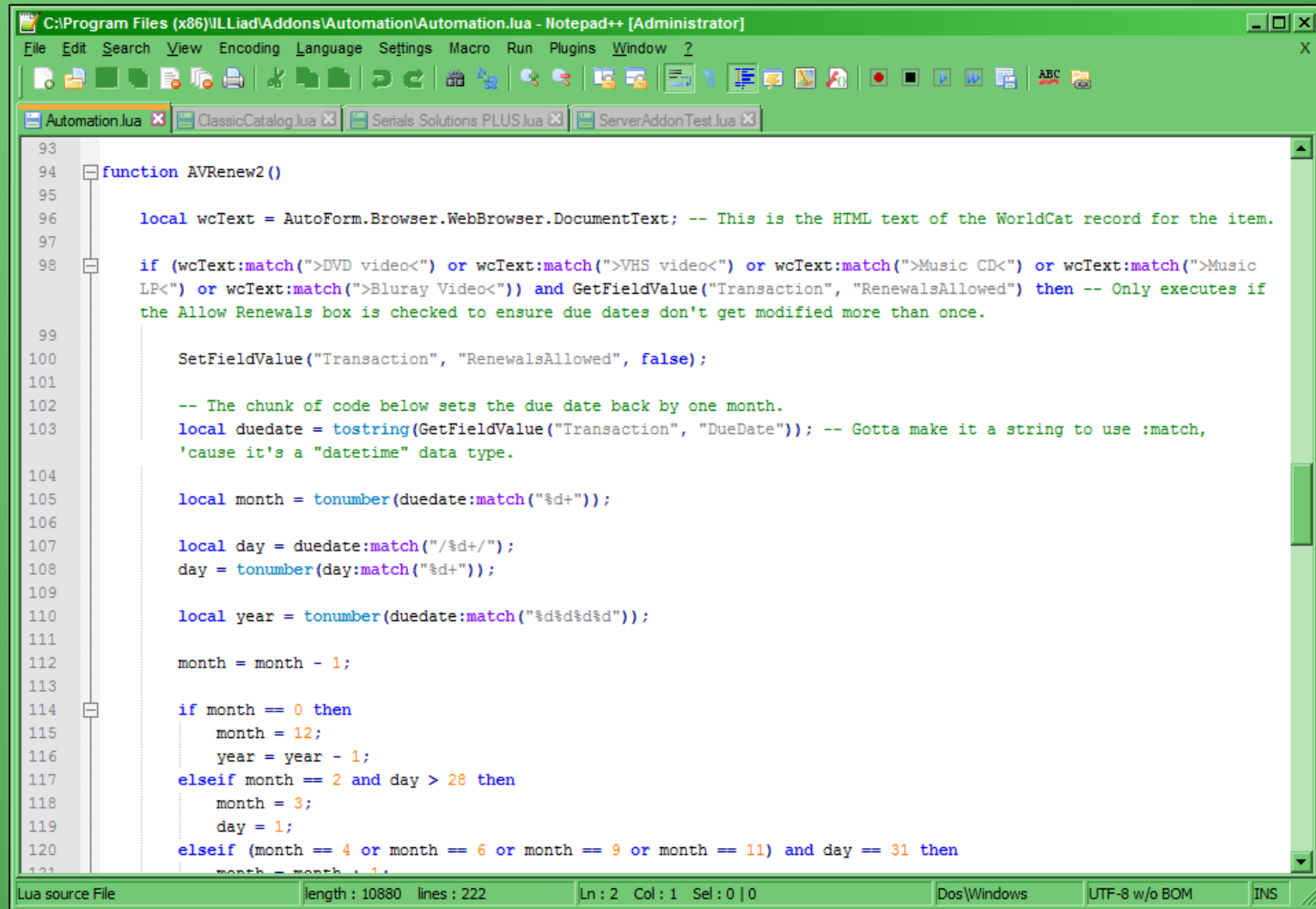


DIY Addons for ILLiad



The screenshot shows a Notepad++ window titled "C:\Program Files (x86)\ILLiad\Addons\Automation\Automation.lua - Notepad++ [Administrator]". The window contains a Lua script with the following code:

```
93
94 function AVRenew2 ()
95
96     local wcText = AutoForm.Browser.WebBrowser.DocumentText; -- This is the HTML text of the WorldCat record for the item.
97
98     if (wcText:match(">DVD video<") or wcText:match(">VHS video<") or wcText:match(">Music CD<") or wcText:match(">Music
99     LP<") or wcText:match(">Bluray Video<")) and GetFieldValue("Transaction", "RenewalsAllowed") then -- Only executes if
100     the Allow Renewals box is checked to ensure due dates don't get modified more than once.
101
102     SetFieldValue("Transaction", "RenewalsAllowed", false);
103
104     -- The chunk of code below sets the due date back by one month.
105     local duedate = tostring(GetFieldValue("Transaction", "DueDate")); -- Gotta make it a string to use :match,
106     'cause it's a "datetime" data type.
107
108     local month = tonumber(duedate:match("%d+"));
109
110     local day = duedate:match("/%d+/");
111     day = tonumber(day:match("%d+"));
112
113     local year = tonumber(duedate:match("%d%d%d%d"));
114
115     month = month - 1;
116
117     if month == 0 then
118         month = 12;
119         year = year - 1;
120     elseif month == 2 and day > 28 then
121         month = 3;
122         day = 1;
123     elseif (month == 4 or month == 6 or month == 9 or month == 11) and day == 31 then
124         month = month + 1;
```

The status bar at the bottom of the window displays: "Lua source File", "length: 10880 lines: 222", "Ln: 2 Col: 1 Sel: 0 | 0", "Dos\Windows", "UTF-8 w/o BOM", and "INS".

What This Presentation Covers:

- The basic function and structure of client-side addons
- Resources for creating and editing client-side addons
- Examples of how to use client-side addons to streamline workflows and add functionality

What This Presentation Does Not Cover:

- Server addons (sorry, I only have an hour!)
- System-level addons
- How to become fluent in Lua

Types of Addons

Client

- Runs when a transaction is opened
- Usually, but not always, a tab in the transaction
- Examples: WorldCat Local Search, Serials Solutions Citation Linker, CCC GetItNow

System

- Usually runs when a select few actions are performed on a transaction (marked found, checked in from lender, etc.)
- Can also be used to display an addon tab on the ILLiad home screen
- Examples: IDS NCIP Client, Workflow Toolkit

Server

- Run at set intervals in the background—requires no manual trigger or intervention.
- Uses SQL queries to affect multiple transactions at once.
- Examples: Electronic Delivery Reminder, Twilio SMS Notifications

The Possibilities...

Search your library's OPAC automatically!

The screenshot shows a web browser window displaying the ECU Libraries OPAC. The browser's address bar shows the URL "520284 - Borrowing Request". The page features a navigation menu with options like "Borrowing Processing", "Printing", "Copyright", "OCLC Request", and "Classic Catalog". The main content area displays the search results for the ISBN "9780670813643".

ECU Libraries

Search Begins with (Alphabetical Browse) Call Number Ask a Librarian Course Reserves Digital Collections Floor Maps Search Tips My Account

Go Back New Search Change Display Saved Link to Page

record 1 of 1 for search ISBN/ISSN "9780670813643" [Change Display](#)

Continue search in
[WorldCat](#)

Requests
[Interlibrary Loan Request](#)
[Recall Request](#)
[Rush Catalog Request](#)
[Purchase Request](#)

Item Details
 Save



Item Information [Catalog Record](#)

Title: Misery
Author: King, Stephen, 1947-
Publication Info: New York, N.Y., U.S.A. : Viking, ©1987.
Description: 310 pages ; 25 cm
ISBN: 0670813648
Item info: 2 items available at Joyner Library @ ECU.

Holdings [Change Display](#)

Joyner Library @ ECU

Call Number	Copy Material	Location
PS3561.I483 M5 1987 2	1 BOOK	Joyner Stacks
PS3561.I483M5 1987	1 BOOK	Joyner Stacks

Cancelled by ILL Staff Borrowing

The Possibilities...

Modify existing addons to suit your needs!

The screenshot shows a web browser window with a toolbar at the top containing various navigation and utility icons. The browser's address bar and tabs are visible, with the current page being a ScienceDirect article. The article title is "Why we sleep: the evolutionary pathway to the mammalian sleep" by M.C. Nicolau et al., published in "Progress in Neurobiology" (Volume 62, Issue 4, November 2000, pages 379-406). The ScienceDirect logo and navigation options are at the top of the page content. A sidebar on the right lists recommended articles and citing articles. At the bottom of the browser window, a status bar indicates "Request Finished" and "Borrowing".

Request Finished

Borrowing

ECU Libraries

The Possibilities...

Automate processes behind the scenes!

The screenshot shows the 'Manage Addons' application window. At the top, there is a 'Home' tab and a toolbar with buttons for 'Reset Cache', 'Save Settings', 'Reset Settings', and 'Create Connection File'. Below the toolbar is a table listing addons. The 'Automation' addon is selected, and its details are shown in a panel below the table.

Name	Author	Version	Active	Description
Amazon Book Searches	Atlas Systems, Inc.	1.1	<input type="checkbox"/>	Performs an Amazon search for the LoanTitle for loans.
Automation	Andrew Morgan	1.6.25	<input checked="" type="checkbox"/>	Automates stuff: <ul style="list-style-type: none">• Unchecks the Allow Renewals box and reduces the due date by one month for audiovisual items.• Controls the Max Cost routing rules.• Clears the Reason For Cancellation field for items whose conditions have been accepted.• Checks the patron's status in Symphony for Borrowing requests in Awaiting Request Processing and pops up a message if they're not OK. Also pops up a message if the patron's Banner ID is missing or improperly formatted and copies the patron's name in the "last name, first name" format to ease lookup in Symphony.• Pops up a message for Library Use Only items that have been granted a renewal to remind us to write the new due date on the item's slip up front.• Saves the transaction upon opening. More functions can be added as needed. AVRenew, FigureCost, and CancelClear only run on Lending requests. CopyID and LibUseRenew only run on Borrowing requests.

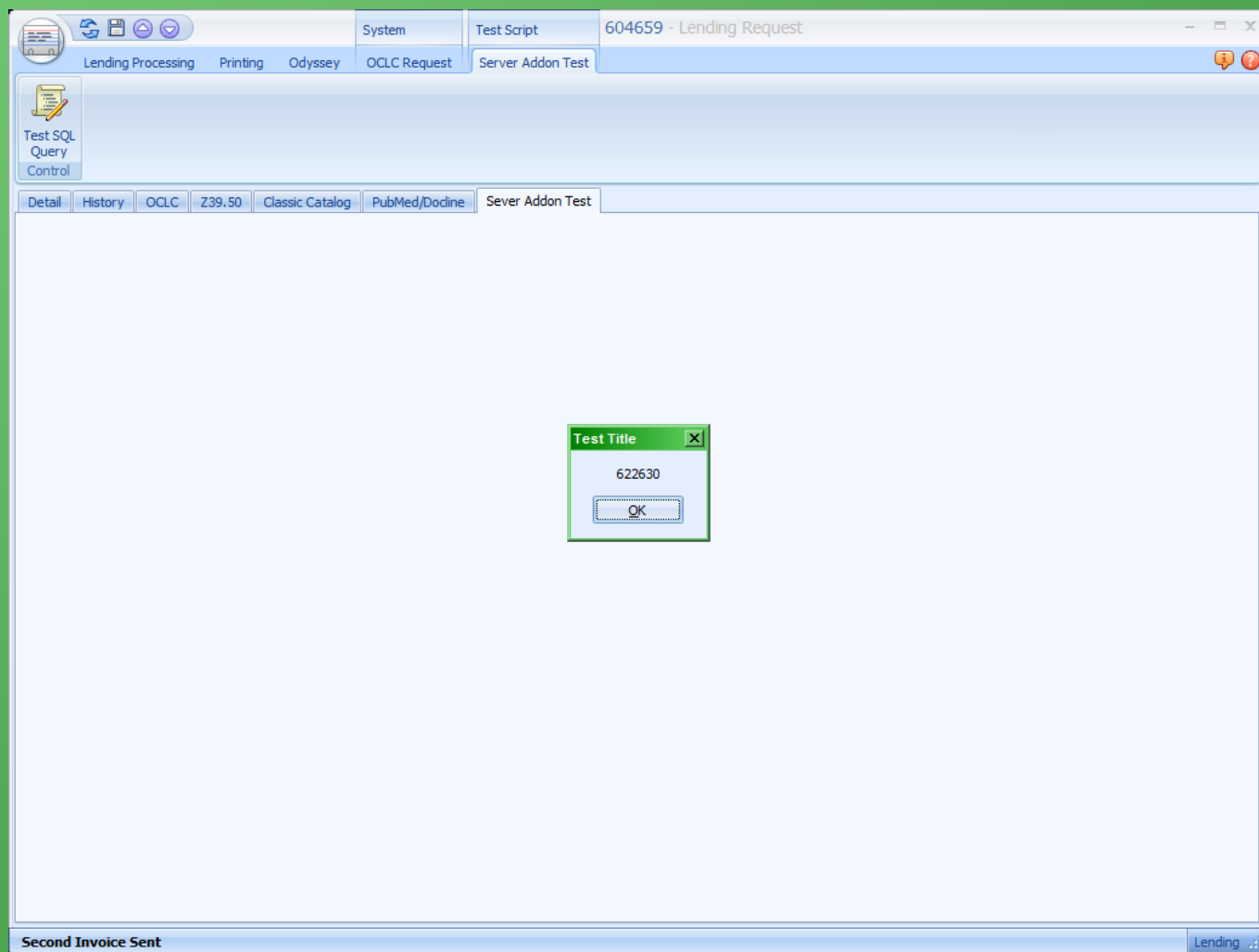
Name	Author	Version	Active	Description
Automation	Andrew Morgan	1.6.25	<input checked="" type="radio"/> Yes <input type="radio"/> No	Automates stuff: <ul style="list-style-type: none">• Unchecks the Allow Renewals box and reduces the due date by one month for audiovisual items.• Controls the Max Cost routing rules.• Clears the Reason For Cancellation field for items whose conditions have been accepted.• Checks the patron's status in Symphony for Borrowing requests in Awaiting Request Processing and pops up a message if they're not OK. Also pops up a message if the patron's Banner ID is missing or improperly formatted and copies the patron's name in the "last name, first name" format to ease lookup in

Setting Na...	Value	Type	Description
AVRenew	<input checked="" type="checkbox"/>	boolean	Automatically unchecks the Allow Renewals box and reduces the due date by one month for audiovisual items.
FigureCost	<input checked="" type="checkbox"/>	boolean	Makes the Max Cost routing rules work.
CancelClear	<input checked="" type="checkbox"/>	boolean	Automatically clears Reason for Cancellation field for items in Awaiting Conditional Request Processing.
PatronStatus	<input checked="" type="checkbox"/>	boolean	Automatically checks the patron's status for requests in Awaiting Request Processing. If a patron's Banner ID is improperly formatted or missing, pops up an alert and copies the patron's name to the clipboard.
LibUseRenew	<input checked="" type="checkbox"/>	boolean	Pops up a message for Library Use Only items that have been granted a renewal to remind us to write the new due date on the item's slip up front.
AutoSave	<input checked="" type="checkbox"/>	boolean	Saves the transaction automatically when it is opened (prevents that annoying "Do you want to save?" dialog when you haven't done anything to a request). This is highly recommended if you're using any of the other Automation functions.

Cache reset.

The Possibilities...

Test server addons!



This is more exciting than a still image can convey. Trust me.

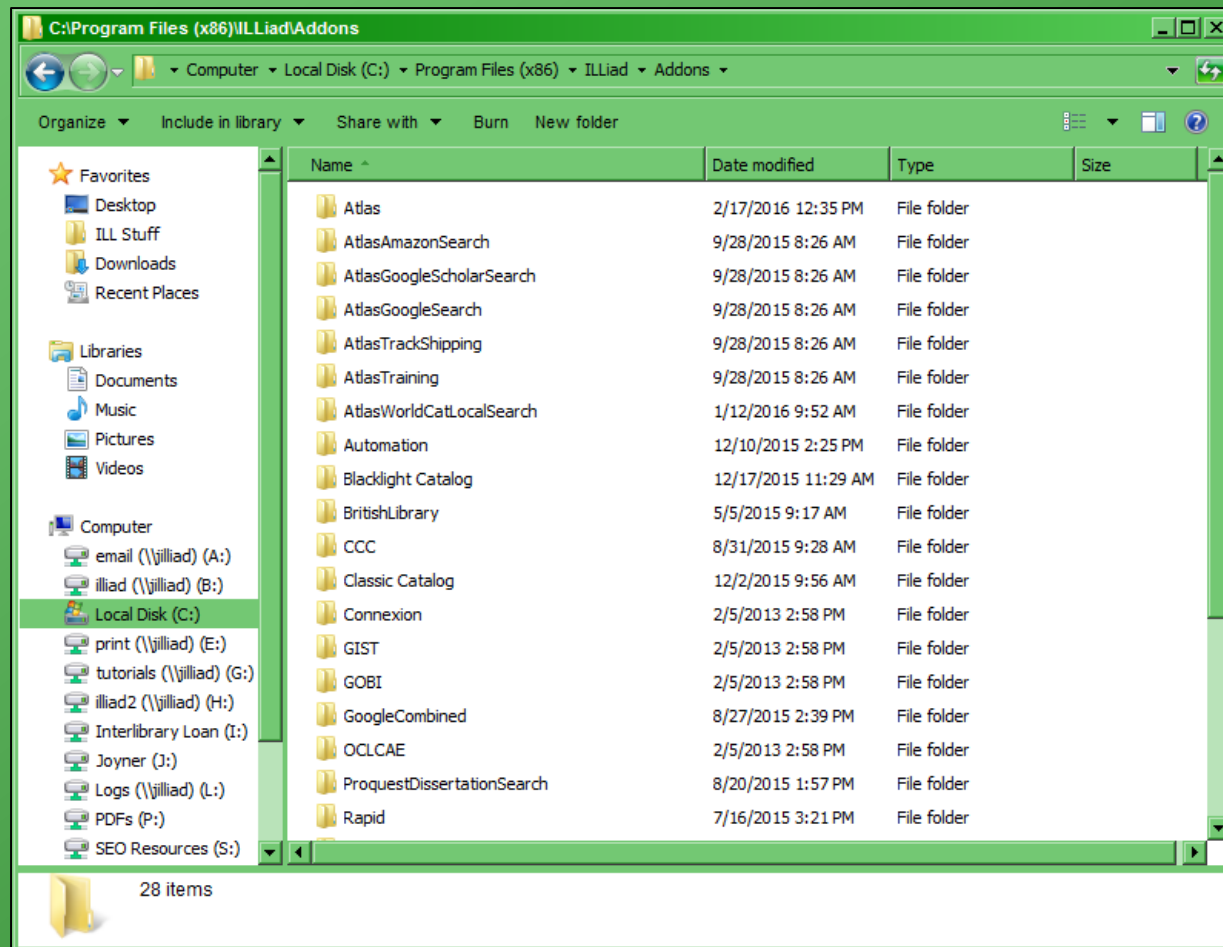
What You'll Need

- ~~1/4 cup of butter~~
- Notepad++
- Lua reference manual
 - Lua demo
- ILLiad addons documentation
 - Key data tables
- ILLiad addons directory

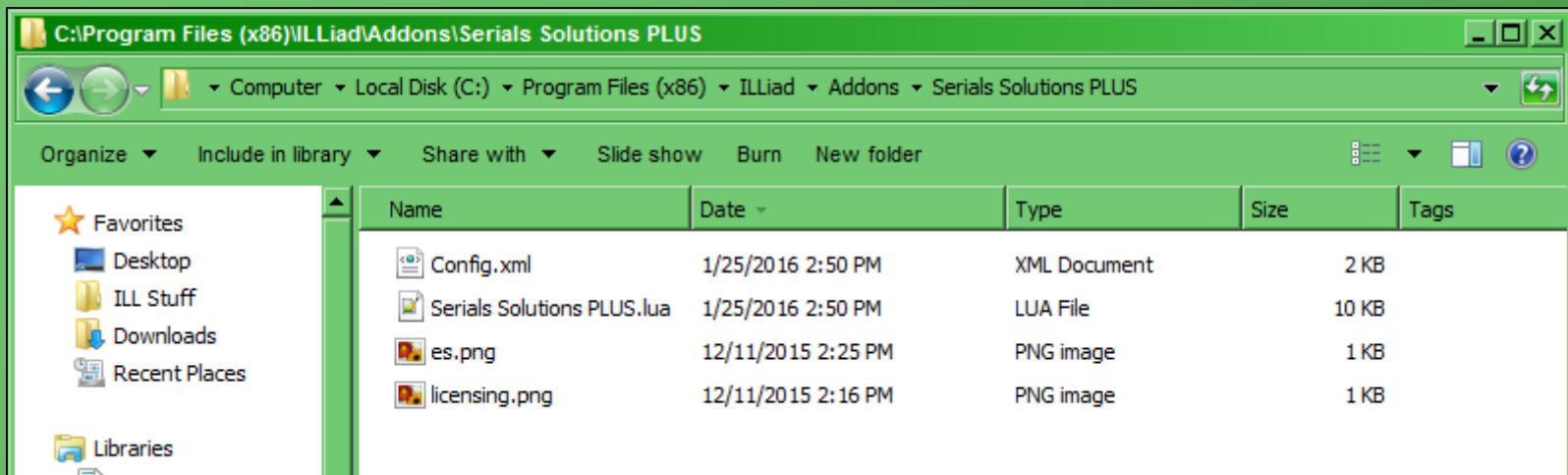
Getting Started

Try not to break anything.

- If your site uses a shared addons directory, you should always test on your computer first.
- The file path for addons is C:\Program Files\ILLiad\Addons. Make a shortcut to it.
 - Program Files (x86) for 64-bit operating systems after Vista



Every client addon is composed of at least two files:



- The Lua file contains the code that is executed when the transaction is opened.
- The XML file contains the settings that can be adjusted in the Manage Addons section of the client.
- Image and layout files may also be in this folder.

Basic Structure

Client addons all (mostly) share the same basic structure:

```
1  -- Addon version and revision notes (optional, but helpful), commented out.
2
3  local settings = {};
4  settings.Setting1 = GetSetting("Setting1");
5  settings.Setting2 = GetSetting("Setting2");
6
7  local interfaceMgr = nil;
8  local whateverForm = {};
9
10 whateverForm.Form = nil;
11 whateverForm.Browser = nil;
12 whateverForm.RibbonPage = nil;
13
14 require "Atlas.AtlasHelpers"; -- And any other libraries or files you need.
15
16 function Init() -- All addons need an Init function.
17
18     interfaceMgr = GetInterfaceManager(); -- Gets the Interface Manager; vital for creating the addon interface.
19
20     whateverForm.Form = interfaceMgr.CreateForm("Whatever", "Script"); -- Creates the addon form. This is essentially everything that appears within the addon tab.
21
22     whateverForm.Browser = whateverForm.Form.CreateBrowser("Whatever", "Whatever Browser", "Whatever Ribbon"); -- Creates the browser for your addon, so you can navigate to web pages.
23
24     whateverForm.Browser.TextVisible = false; -- Hides the annoying label for the browser. Trust me, it takes up a lot of space. You always want this hidden.
25
26     whateverForm.Browser.WebBrowser.ScriptErrorsSuppressed = true; -- Suppresses Javascript errors on web pages. Always best to have this on, especially if you're dealing with WorldCat.
27
28     whateverForm.RibbonPage = whateverForm.Form.GetRibbonPage("Whatever Ribbon"); -- Creates the ribbon above the tab; this is where your buttons go.
29
30     whateverForm.RibbonPage.CreateButton("Pop-up!", GetClientImage("Wizard32"), "PopupFunction", "Button Space"); -- This is a button with the text "Pop-up!" It triggers the PopupFunction function, and is located in the Button
31     Space group.
32
33     whateverForm.Form:Show(); -- Makes the form visible. Usually you want that.
34
35     MainFunction(); -- Runs MainFunction, which is defined below.
36
37 end
38
39 function MainFunction()
40
41     whateverForm.Browser:Navigate("http://www.atlas-sys.com"); -- Navigates to the Atlas website.
42
43 end
44
45 function PopupFunction() -- Runs when the button is pressed;
46
47     interfaceMgr.ShowMessage("Pop-up message: You pressed the button. Congratulations.", "Pop-up title: DING!");
48
49 end
```

Basic Structure - Settings

Settings are typically declared in the first lines of the addon. These are special variables that can be changed by users from the Manage Addons section of the client.

```
3  local settings = {};  
4  settings.Setting1 = GetSetting("Setting1");  
5  settings.Setting2 = GetSetting("Setting2");  
6
```

Basic Structure – Form(s)

The `InterfaceManager` and any forms your addon uses are initialized next. The form includes everything in the tab created by the addon, whether it's visible or not.

```
6
7     local interfaceMngr = nil;
8     local whateverForm = {};
9
10    whateverForm.Form = nil;
11    whateverForm.Browser = nil;
12    whateverForm.RibbonPage = nil;
13
```

Basic Structure – Other Files (optional)

You may use the command "require" to allow the addon to access information in other files, such as the AtlasHelpers library.

```
13  
14     require "Atlas.AtlasHelpers";  
15
```

Basic Structure – Init function

This is the function that runs when the transaction is opened.

```
15  
16 function Init()  
17  
18     interfaceMngr = GetInterfaceManager();  
19  
20     whateverForm.Form = interfaceMngr.CreateForm("Whatever", "Script");  
21  
22     whateverForm.Browser = whateverForm.Form.CreateBrowser("Whatever", "Whatever Browser", "Whatever Ribbon");  
23  
24     whateverForm.Browser.TextVisible = false;  
25  
26     whateverForm.Browser.WebBrowser.ScriptErrorsSuppressed = true;  
27  
28     whateverForm.RibbonPage = whateverForm.Form.GetRibbonPage("Whatever Ribbon");  
29  
30     whateverForm.RibbonPage.CreateButton("Pop-up!", GetClientImage("Wizard32"), "PopupFunction", "Button Space");  
31  
32     whateverForm.Form.Show();  
33  
34  
35     MainFunction();  
36  
37 end
```


Basic Structure – Other Functions

Just because you *can* cram everything into the Init function doesn't mean you *should*.

```
39  function MainFunction()  
40  
41      whateverForm.Browser:Navigate("http://www.atlas-sys.com");  
42  
43  end  
44  
45  function PopupFunction()  
46  
47      interfaceMgr.ShowMessage("Pop-up message: You pressed the button. Congratulations.", "Pop-up title: DING!");  
48  
49  end
```


Basic Structure – Comments

Two hyphens denote a comment. Comments are not executed, but are helpful for others who may read your code.

```
15  
16   function Init() -- All addons need an Init function.  
17
```

Basic Structure – Demo

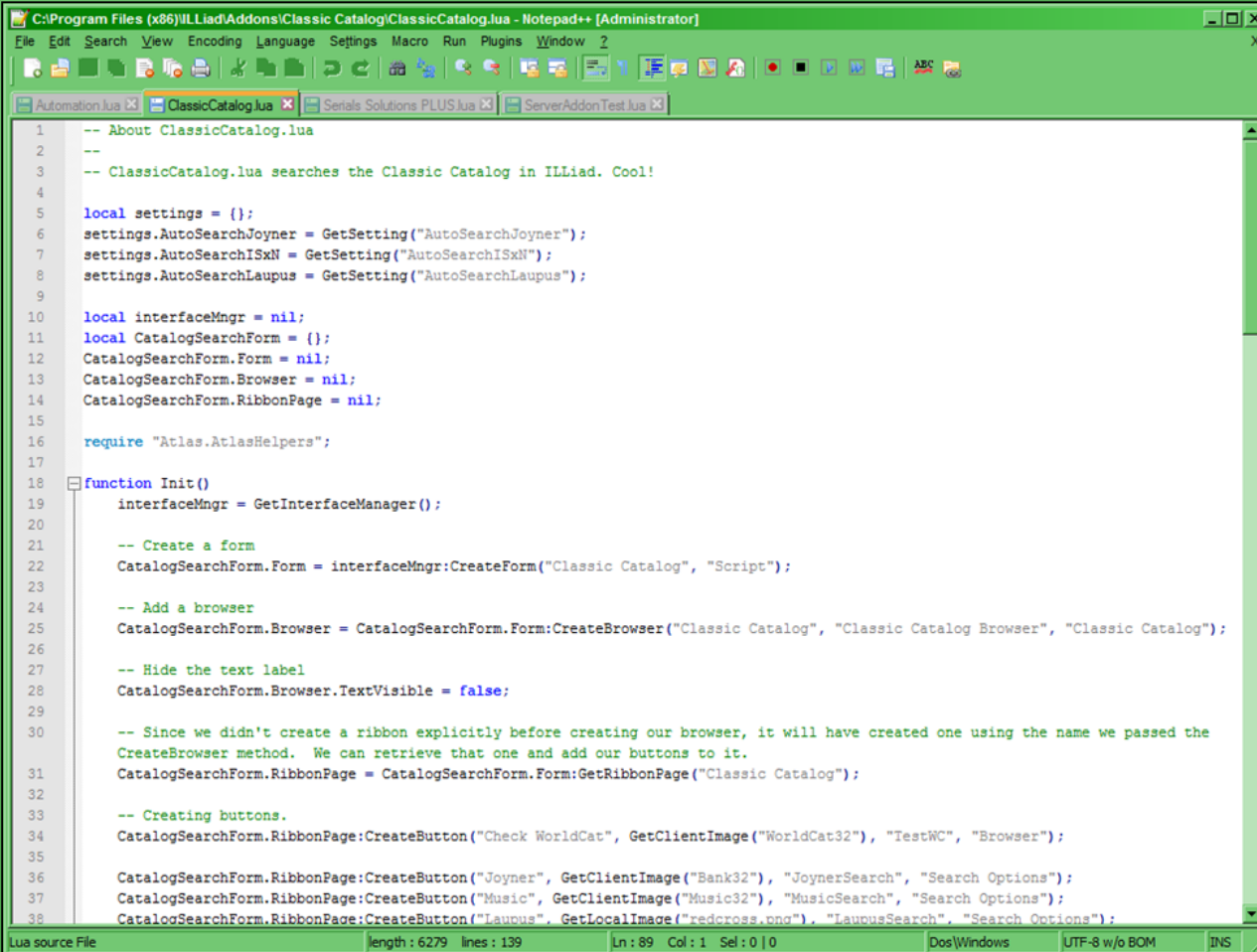
Here's the result of the code in the previous slides:



The screenshot shows a web browser window displaying the Atlas Systems website. The browser's address bar shows "Script" and "622630 - Lending Request". The website header includes the Atlas Systems logo and navigation links: PRODUCTS, SERVICES, ILLIAD CONFERENCE, TRAINING, DOCUMENTATION, SUPPORT, and ABOUT. A central banner features the "ares automating reserves" logo and three call-to-action buttons: "Sign up for a web demo", "Tools for current licensees", and "Contact Us". A pop-up message box is overlaid on the banner, displaying the text: "Pop-up title: DING! Pop-up message: You pressed the button. Congratulations." Below the banner, there are three columns of text describing services: "Aeon: Managing Special Collections", "Ares: Processing Reserves", and "Get The Latest News from Atlas Systems". The browser's status bar at the bottom shows "Awaiting Secondary Label Printing" and "Lending ...".

Basic Structure – Easy Template

If this structure doesn't seem so basic, don't worry. You can always use existing addons as templates.



```
1  -- About ClassicCatalog.lua
2  --
3  -- ClassicCatalog.lua searches the Classic Catalog in ILLiad. Cool!
4
5  local settings = {};
6  settings.AutoSearchJoyner = GetSetting("AutoSearchJoyner");
7  settings.AutoSearchISxN = GetSetting("AutoSearchISxN");
8  settings.AutoSearchLaupus = GetSetting("AutoSearchLaupus");
9
10 local interfaceMgr = nil;
11 local CatalogSearchForm = {};
12 CatalogSearchForm.Form = nil;
13 CatalogSearchForm.Browser = nil;
14 CatalogSearchForm.RibbonPage = nil;
15
16 require "Atlas.AtlasHelpers";
17
18 function Init()
19     interfaceMgr = GetInterfaceManager();
20
21     -- Create a form
22     CatalogSearchForm.Form = interfaceMgr.CreateForm("Classic Catalog", "Script");
23
24     -- Add a browser
25     CatalogSearchForm.Browser = CatalogSearchForm.Form.CreateBrowser("Classic Catalog", "Classic Catalog Browser", "Classic Catalog");
26
27     -- Hide the text label
28     CatalogSearchForm.Browser.TextVisible = false;
29
30     -- Since we didn't create a ribbon explicitly before creating our browser, it will have created one using the name we passed the
31     CreateBrowser method. We can retrieve that one and add our buttons to it.
32     CatalogSearchForm.RibbonPage = CatalogSearchForm.Form.GetRibbonPage("Classic Catalog");
33
34     -- Creating buttons.
35     CatalogSearchForm.RibbonPage.CreateButton("Check WorldCat", GetClientImage("WorldCat32"), "TestWC", "Browser");
36
37     CatalogSearchForm.RibbonPage.CreateButton("Joyner", GetClientImage("Bank32"), "JoynerSearch", "Search Options");
38     CatalogSearchForm.RibbonPage.CreateButton("Music", GetClientImage("Music32"), "MusicSearch", "Search Options");
39     CatalogSearchForm.RibbonPage.CreateButton("Laupus", GetLocalImage("redcross.png"), "LaupusSearch", "Search Options");
```

Lua source File length: 6279 lines: 139 Ln: 89 Col: 1 Sel: 0 | 0 Dos/Windows UTF-8 w/o BOM INS

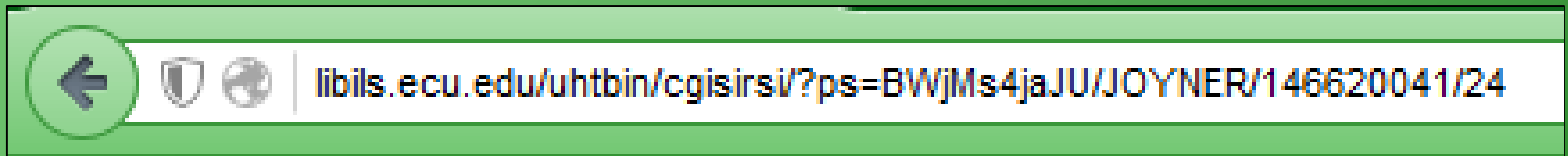
If-Then Statements

If-then statements, particularly in conjunction with `GetFieldValue`, are incredibly useful.

```
32 function Init()
33     local reqType = GetFieldValue("Transaction", "RequestType");
34     local process = GetFieldValue("Transaction", "ProcessType");
35     local queue = GetFieldValue("Transaction", "TransactionStatus");
36     local nvtgc = GetFieldValue("User", "NVTGC");
37
38     interfaceMgr = GetInterfaceManager();
39
40     if process == "Lending" then -- Ensures that these functions only run on Lending requests.
41         if settings.AVRenew and queue == "Awaiting Lending Request Processing" and reqType == "Loan" then
42             --AVRenew only runs on loans in Awaiting Lending Request Processing.
43             AVRenew();
44         end
45     end
46
47     if settings.FigureCost then
48         FigureCost();
49     end
50
51     if settings.CancelClear and queue == "Awaiting Conditional Request Processing" then
52         SetFieldValue("Transaction", "ReasonForCancellation", nil);
53     end
54
55 end
```

Page Handlers

So, you want to make an addon for your OPAC, but your OPAC doesn't use nice, easily manipulated URLs?



That's where page handlers come in handy. That's a link. Click it to read the page handlers documentation.

Page Handlers – Pick One

The type of page handler you use will depend on the page. When in doubt, there's always the custom page handler.

```
170
171 if not banner:match("8%d%d%d%d%d%d%d") then
172     interfaceMgr:ShowMessage("Improperly formatted or missing Banner ID.", "Patron needs Banner ID!");
173     clipboard.SetData("Text", GetFieldValue("User", "LastName") .. ", " .. GetFieldValue("User", "FirstName"));
174 else
175     StatusForm.Browser:RegisterPageHandler("custom", "StatusCheck", "PatronStatus3", false);
176     StatusForm.Browser:Navigate("http://libils.ecu.edu:8080/svms/rest/patron/lookupPatronInfo?clientID=ECULib
177 end
178
179 end
180
181 function PatronStatus3()
182     local docText = StatusForm.Browser.WebBrowser.DocumentText;
183     local status = docText:match("<statusType>.+</statusType>"):gsub("< /*statusType>", "");
184
185     if status ~= "OK" then
186         interfaceMgr:ShowMessage("Patron is " .. status .. ".", "This patron is not okay!");
187     end
188 end
189
190 function LoginCheck()
191     local docText = StatusForm.Browser.WebBrowser.DocumentText;
192
193     if docText:match("LoginUserResponse") then
194         return true;
195     else
196         return false;
197     end
198 end
199
200 function StatusCheck()
201     local docText = StatusForm.Browser.WebBrowser.DocumentText;
202
203     if docText:match("LookupPatronInfoResponse") then
204         return true;
205     else
206         return false;
207     end
208 end
209
```

Page Handlers – Dig Into HTML

Page handlers are pretty straightforward, but you'll have to take a look at the HTML of the target page first.

```
view-source:http://libils.ecu.edu/uhtbin/cgisirsi/x/SIRSI/0/57/60/1178/X?user_id=ECUWEBCAT
196
197 <li class="menu_link"><a href="javascript:show_permalink('http://libils.ecu.edu/uhtbin/cgisirsi/x/SIRSI/0/57/60/1178/X?user_id=ECUWEBCAT')">Link to Page</a></li>
198
199
200 <li class="menu_link"><a href="/uhtbin/cgisirsi/?ps=JDFmhK9jFa/JOYNER/323560048/163" title="" target="_top"></a></li>
201 <li class="menu_link"><a name="skipnav"></a></li>
202 </ul>
203 </div>
204
205 <div class="columns_container">
206 <div class="column pct60 left_column">
207 <div class="content_container quick_search">
208 <div class="content">
209 <!-- Some complication is added here because of the "more searches" button having to be inside of the "icon search" form. So, if this is an "icon search, start the f
210 <div class="searchservices">
211 <!-- Print element's description if present, else print the heading -->
212 <h3>Begins with (Alphabetical Browse)</h3>
213 <!-- Catalog Access destinations may either be of type Search, Browse, or Term Search -->
214 <!-- Copyright (c) 1996 - 2010, SirsiDynix - Catalog Access - Browse form (for state #3). -->
215
216 <!-- Labelled fields -->
217 <!-- Copyright (c) 1996 - 2010, SirsiDynix - Catalog Access - Browse form (for state #3). -->
218 <form name="searchform" method="post" action="/uhtbin/cgisirsi/?ps=G5y0sxxXVY/JOYNER/323560048/24">
219 <table border="0" cellpadding="5" cellspacing="0" align="center" class="searchcontent">
220 <tr>
221 <!-- Copyright (c) 2000 - 2010, SirsiDynix - Generates one field select list for the "Advanced-type" search form (for state #3). This page file must be included with
222 must be set to the ENTRYNUMCD value of this field, and DEFAULT_FIELD to the default label of this field. -->
223 <!-- If previously submitted a search, then SRCH_LABEL#, SRCH_KIND#, etc. will contain the type of search for the '#' field, where '#' is the ENTRYNUM variable. If t
224 then use them; otherwise, use the defaults from the server's response list. -->
225
226 <select name="srchfield1" id="srchfield1" title="Limit By">
227 <option value="TI^TITLE^SERIES^Title Processing^title">title</option>
228 <option value="AU^AUTHOR^AUTHORS^Author Processing^author">author</option>
```

Pattern Matching

Pattern matching is useful for creating custom page handlers as well as functions for automatically correcting or formatting data.

```
66 function Class.ISxNFix(str) -- This makes sure the ISxN is formatted correctly. Removes hyphens from
    ISBNs and cuts off extra characters, cuts off additional ISSNs, and formats ISSNs with a hyphen if they
    don't already have one.
67
68     str = str:gsub("-", "");
69
70     if str:match("%d%d%d%d%d%d%d%d%d%d%w+") then
71         str = str:match("%d%d%d%d%d%d%d%d%d%d%w+");
72     elseif str:match("%d%d%d%d%d%d%d%w") then
73         str = str:match("%d%d%d%d%d%d%d%w");
74         str = str:sub(1, 4) .. "-" .. str:sub(5, 8);
75     end
76
77     return str;
78 end
```


Config Files

Config files are relatively simple. These are where you set the version number and the settings for the addon.

```
3 <Name>Presentation Addon</Name>
4 <Author>Andrew Morgan</Author>
5 <Version>1.0.01</Version>
6 <Active>False</Active>
7 <Type>Addon</Type>
8 <Description>An example addon for my ILLiad Conference 2016 presentation.</Description>
9 <Forms>
10   <Form>FormRequest</Form>
11 </Forms>
12 <Settings>
13   <Setting name="Setting1" value="true" type="boolean">
14     <Description>Setting example 1.</Description>
15   </Setting>
16   <Setting name="Setting2" value="http://www.google.com" type="string">
17     <Description>Setting example 2.</Description>
18   </Setting>
19 </Settings>
20 <Files>
21   <File>Presentation Addon.lua</File>
22 </Files>
23 </Configuration>
```

Questions?

Feel free to e-mail me at morgana@ecu.edu.

