

## MageFX:

Porting Mage over to JavaFX is no easy task, let alone developing Mage itself.

TO DO:

- Server Connection and communication
- proper Card Discovery and implementation
- **DeckEditor** (in process)
- **CollectionView**
- **GamesViewer**
- **BattleField**
- preferences and settings
- and just about everything else ..

I have implemented many components, however many of them also need to be polished and refined or even redesigned.

The Items in Red are the main "windows" or "Panels" to be worked with.

## Features I wish to implement:

- Better user experience ie: totalWins based bonuses, like unlocking more avatars etc..
- Turn based Animated Timers ( no more waiting for opponent who had to pee, this makes you PLAY or lose )
- the Better User experience covers the rest...

I would like and DO need some People to help in the following areas:

- A CSS Guru to implement/write the Themes (the base template will be provided at the bottom)
- Someone to help with Layout's/component design
- Graphics Guru for button Icons and other graphical elements

(I can do all these things, but I'll be wearing Diapers again before I get done)

## Links and References:

**Java 7 update 21** is the JDK being used for development.

Link to the Project: [MageFX-Client Project](#)

Link to JavaFX-CSS guide: [-FX-CSS-Reference Guide](#)

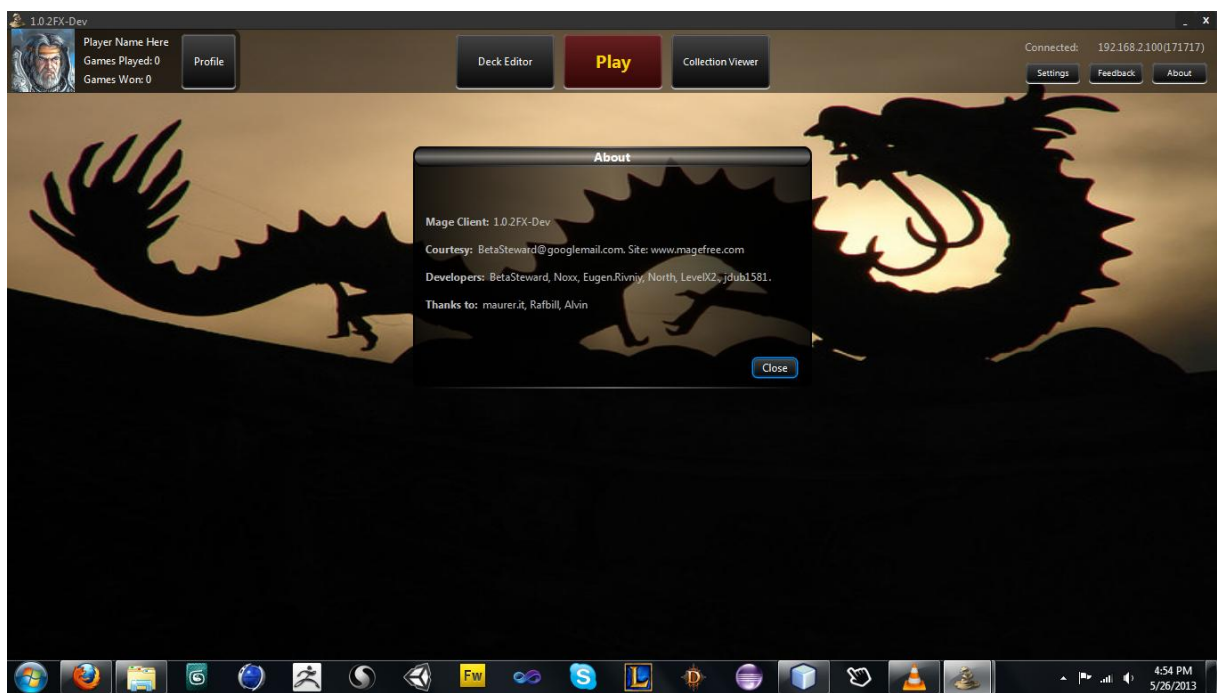
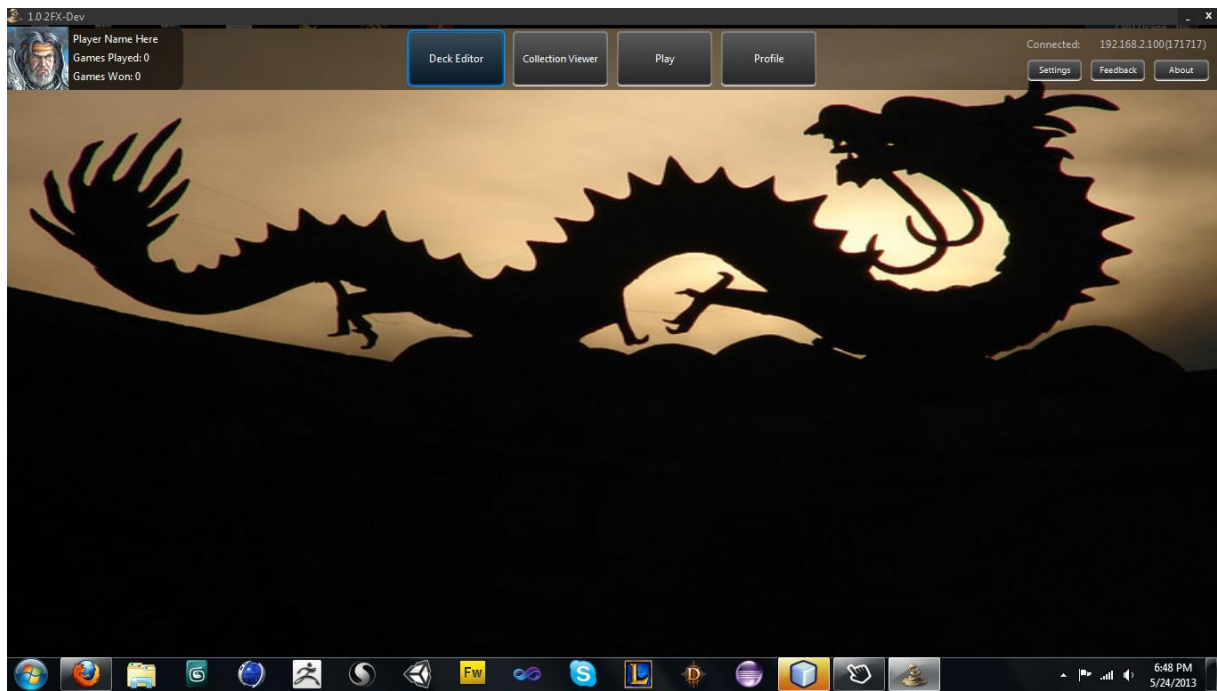
Link to SceneBuilder: [SceneBuilder 1.1](#)

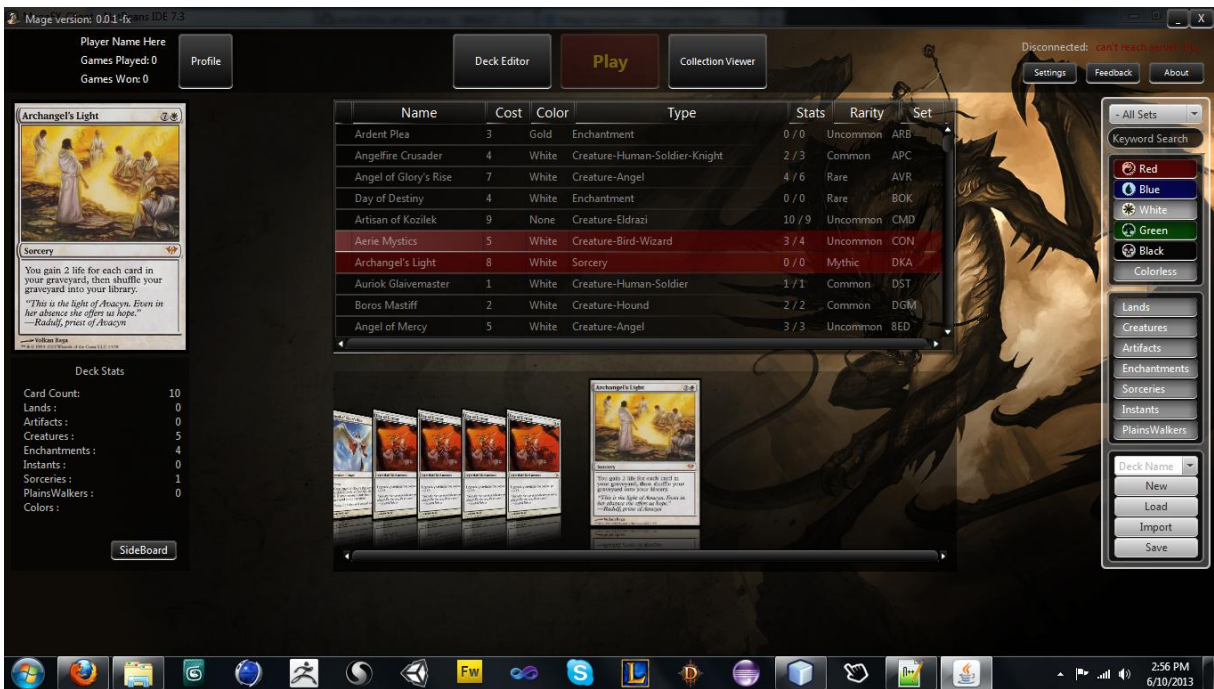
I am attaching the "Default" CSS file extracted from jfxrt.jar

Pretty much the only thing that needs to be changed for Themes are the elements under .root{ }

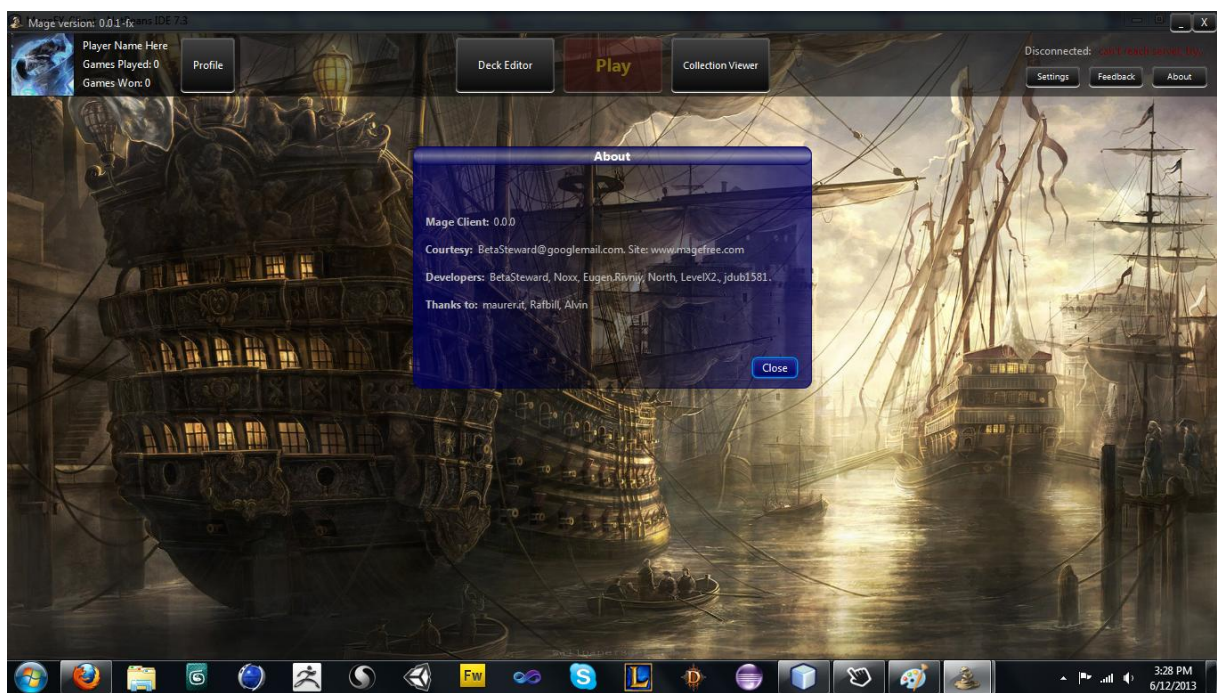
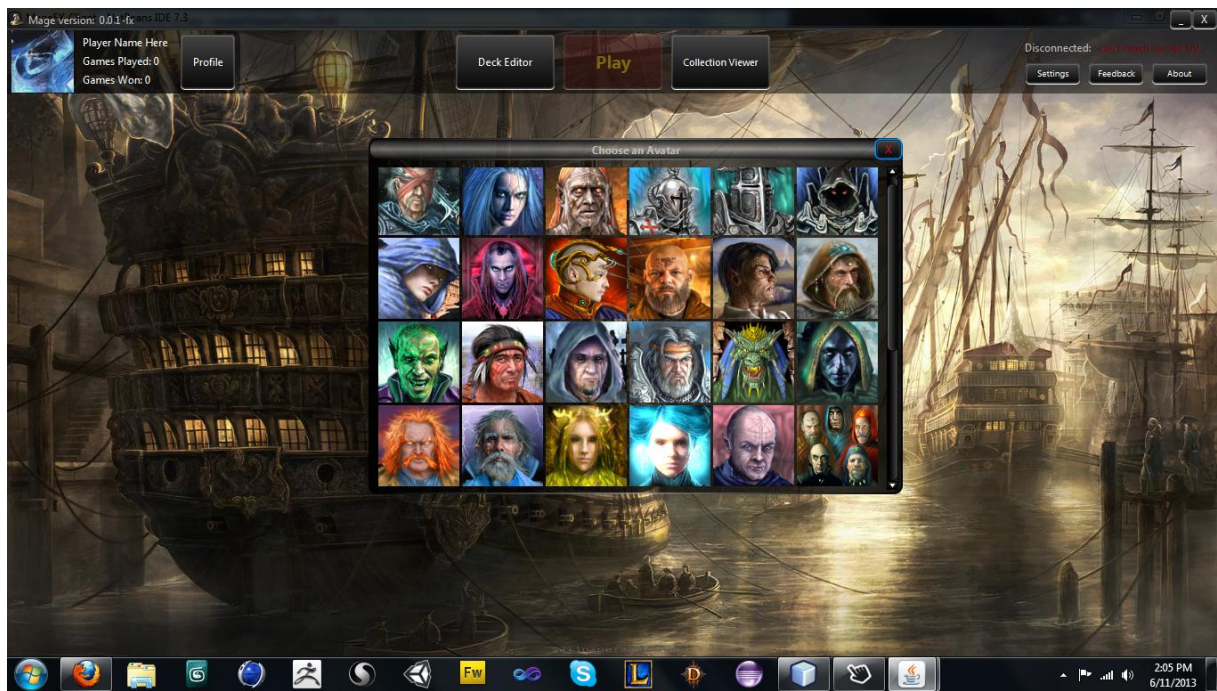
There are a few exceptions that will need to be changed manually for transparency if needed.

Evolution in Screen Shots ... (last image shows an implementation of my Themeable interface on the AboutDialog)









# MageFX8

Posteingang x



**jdub1581** <jdub1581@gmail.com> 04.04.14

an mage-developers

Ok guys, transferred most files over to the new Project...  
Still need to fix many of the classes...  
Right away I noticed TrueZip is not used any more.. is that true?

So this may take a while to go thru and find what's still in the project ..

On another note, I have been playing with the Thread merge (Swing and FX), and I can confirm that updates to FX UI can be made from Swing threads now..

this is the command to get it going.. : -Djavafx.embed.singleThread=true    still experimental, but I dont see a problem if it works

This will help a lot without having to convert threads to jfx..

Anyways, I will be creating a bunch of blank classes with descriptors explaining what the class should do for those who want to contribute... may take a day or 2, but it will get done!



**North** <robyter@gmail.com> 05.04.14

an mage-developers

Why do we need to have a single thread for Swing and FX? If we rewrite the thing, there is no point in using any Swing code at all. What were the Swing components that you think are needed here?



**North** <robyter@gmail.com> 05.04.14

an mage-developers

JDK 7 has support for zip files as virtual file systems so the need for True Zip vanished.



**jdub1581** <jdub1581@gmail.com> 05.04.14

an mage-developers

Well If I remember right, When I was creating the DeckEditor, and a couple other things, I came accross some WorkerThreads that were hard to convert to JFX Service. As I wrote this I went into the Mage Common src, and It does look like a lot has changed... I haven't jumped into the client src yet but I believe that is where I was running into the problems before. Right away in the Commons, ActionCallback depends on AWT for mouseEvents, and Some "Views" before had Swing dependencies. So the thread merge for me isn't so much about reusing components, but being able to reuse some threads that are already created.

Today I am going to work on getting imported classes updated.. Seems like when I re-pulled Mage src, It is holding some old dependencies like trueZip, so I'm going to delete and clone again and go from there. and the thread merge could be a moot point, but available IF NEEDED..



**robyter@gmail.com** 05.04.14

an mage-developers

I started working on a implementation for the server communication. It will basically be an observable object where you register handlers for all the events that the server is sending. This avoids the god classes and the immense complexity of doing everything into the same place.

I started my own project for this and I will push it on github as soon as I have this in a beta stage (being able to connect to the server and send messages for example).

I also looked at the javafx8 project and I saw that we need a way to handle preferences in a better way. I'll work on this too.

On a side note, the old client looked really good. What I wish to do is extract the theme outside the project so that we don't have any images or resources with copyright. I guess there is quite a bit to do so we'll take it one step at a time.



**jdub1581** <jdub1581@gmail.com> 05.04.14

an mage-developers

as far as Images go, I want to have a base set of images for client Icons, Avatars, Mana Symbols and such can disappear, I just find it easier when doing FXML to have a placeholder so I can See what Im doing rather than an invisible node. I did have the client connected to Server before with Chat working, often tested it and talked to players.. So maybe some of that can help you... Though I di not import any of the FXML controllers that handled that. Check my proj tonight, I should have a bunch done, just waiting on Mage to build now and will jump right in...



**jdub1581** <jdub1581@gmail.com> 05.04.14

an mage-developers

I did do a lot before with the preferences, I divided it up into 4 classes as to not make 1 jumbled up class for everything.  
ie: MagePrefs for client layout and such, ConnectionPrefs for connection, UserPrefs (could merge with MagePrefs), and GamePrefs, for in game settings, and game creation(basically keeps last game type created so pops that up into UI instead of going and setting everything)



**North** <robyter@gmail.com> 05.04.14

an mage-developers

I pushed the server communication module to here: <https://github.com/magefree/fx-client>

This is really simple to understand. Just look in the Main class to see how it is used. You can also take a look at the implementation which is really simple. Ludwig, if you can also take a look it would be great. I promise it won't eat much of your time.

Currently, I'm working on a dependency injection mechanism for the JavaFX client. I know there is one available from Adam Bien but it does not suit our needs. I won't go overboard with engineering this. I expect it to remain as simple as the one we have now. I am tempted to have it annotation based. Here are some solutions:

### **First**

@Inject Session session;

### **Second**

ChatController implements WithSession {

@Override

public void setSession(Session session) { this.session = session; }

}

The first solution can be expanded by adding a name to the variable but I don't know if we'll be injecting different instances of the same class. I expect us to have only one instance for each. At this moment I now I need it for Session and for ObservableServer. I expect to be using this mechanism to inject the preferences too. Apart from this, it will be usable for anything that represents global state. I avoid using singletons as much as possible.

Please let me know what you think.



**North** <robyter@gmail.com> 06.04.14

an mage-developers

I added the Dependency Injection mechanism too. You can see it in action int the MainApp class. I am injecting the "hello world" message.

PS: It feels so good to work on stuff like this.



**jdub1581** <jdub1581@gmail.com> 06.04.14

an mage-developers

Checked it out and seems a better approach than mine... For now, I'm feeling the urge to scrap whet I got and just keep some of the FXML files... Dialogs maybe, deckEditor, gameLobby, and maybe the frame, though I kinda wanna change that up too... Tell you what, If you can work on the backbone stuff like that, I'll get going on the UI.

I'll scrap Images except for GUI related items like appIcon, Mage splash and logo etc..



I'm way out of date with the API 1.0.3 was the last I played with, so can you show me where Image access is done



**robyter@gmail.com** 06.04.14

an mage-developers

I'll be writing a new module for accessing files from zip archives. I think it is best to load the style as a stylesheet. I think you can reference images from the stylesheet. This way there is no custom image loading logic.

I'll be more than happy if you would work on the UI components. I still need to write some code for making the skeleton stable. Can you work on the UI components in a fork for now? We'll pull them into the master branch after I get stuff figured out with JavaFX. I'm really new to it at this point but things are progressing really well.

PS: Your approach was to get it going and you reused what was in the current client. I guess I am too keen on writing stuff differently.



**jdub1581 <jdub1581@gmail.com>** 06.04.14

an mage-developers

Yeah, I was approaching it like I did before, but so much has changed... As far as using css for loading images, in my experience the Images have to be local to the css, in same jar.. Perhaps not as I never tried from an outside package. I do know that an Image can be loaded from any URL (local or web) with little effort, though performance may become an issue with web .. I'll play around with that for a few..

Goal for today is to get The Frame setup, with Subscene, PerspectiveCamera, drag and resizing, After that I want to re-work the DeckEditor.

I made a couple simple additions to your MageObservableServer class here:

```
private final Session session;
private UUID mainRoomId;
private UUID chatId;

public MageObservableServer() {
    this.session = new SessionImpl(this);
}
```

```

public Session getSession() {
    return session;
}

public void connectToServer(Connection con){
    session.connect(con);
    mainRoomId = session.getMainRoomId();
    chatId = session.getRoomChatId(mainRoomId);
    session.joinChat(chatId);
    session.sendChatMessage(chatId, "Greetings from the other world");

    session.disconnect(false);
}

public void disconnectFromServer(){
    session.disconnect(true);
    // not sure if this part is right but I assume so
}

```

I like this approach as It keeps everything together. then Adding getters for UUID's

Also If you get the chance Check out ControlsFX, and JFXtras, and see if any of their components would be a nice fit..



**jdub1581 <jdub1581@gmail.com>** 06.04.14

an mage-developers

Also on a side note, I seem to have trouble building the main Project, I run clean/build from netbeans, it gets to sets, then lags out and eventually fails on BelfrySpirit, running from cmd line mvn clean install, same thing gets to sets then sits there forever with cpu at 100% .. Anyone else experience this or am I missing something? just re-cloned the project yesterday.. src is 1.3.0



**jdub1581 <jdub1581@gmail.com>** 06.04.14

an mage-developers

Ok, pushed the new Frame to git, as far as forking, I really dont use Maven or Git that much other than making things easy to share between my laptop and pc, so I'll just keep adding what I got to my proj, and we can merge them later I spose



**North <robyter@gmail.com>** 07.04.14

an mage-developers

You can remove the mage dependencies when working on the UI. It speeds up your build significantly.

As far as keeping things together. I find it a bad practice that can bite you when you least expect it. I find it more than suitable having them as 2 different objects. One is responsible for sending messages to the server while the other is responsible for notifying you when the server sent a message.

In the same way, I think that those IDs should only be referenced in the contexts that they are used in. Considering we'll have the session and serverObservable in most of our views, I don't see any reason why not to get them when initializing the view.

Regarding those two libraries, I do not know what they provide and we need over what we get by default in JavaFX8.



**North <robyter@gmail.com>** 07.04.14

an mage-developers

I see no problem with this approach. I advise to look at my pom.xml file to see how to improve the build time.



**jdub1581 <jdub1581@gmail.com>** 07.04.14

an mage-developers

Yeah, build time was horrible, removed sets, kept mage-commons though, and the build is now bearable.

I'm still working on the "HomePage" having a little creative block, as far as layout.. I wanna create a "Carousel" for the Menu's, ie: GameLobby, DeckEditor, CollectionViewer, Settings .. And create the components for Server Msgs, and Maybe add the Chat Panel there?

**IF anyone wants to Contribute to Layout Ideas:**

create a FXML file and place a Rectangle with Label(description of what it is) where you think it should go..

copy and paste Code here, or upload to My Project

Not sure how far you got on Properties (leaving that to you), but If you can create a set for GUI that'd be cool,  
I'd like, double-> layoutX & Y, double-> prefWidth & Height, String-> background-URL (for background Img)

I remember before talking about making the client customizable and this is an incredibly easy feature to implement, Img can be local or Web based, By default will be set to the Ship img we have, with an Option under settings to set custom Img via "String".

As far as those Libraries go, JFXtras has for a long time tried to add components that are not available in JFX-core, same with controlsFX.

like JFXtras has a "CirclePane" which will layout child-nodes in a circle, Might be a cool component to have?

<http://jfxtras.org/> pics on site ...

Also ControlsFX has some as well, like Dialogs, and other UI things like an Info-Overlay (for cards maybe?)

[controlsfx](#) links on site ...



**Noxx Mage <magenox@gmail.com>** 28.04.14

an mage-developers

Hi,

I'm really interested in the progress done for the fx client.

Am I right saying that once done such client can be run in browser?

One more question: is the following link the right place to get latest version?

<https://github.com/magefree/fx-client>

Best Regards,

Noxx



**robtyer@gmail.com** 28.04.14



an mage-developers

That is the link indeed. Currently there is only a proof of concept for a main game room chat. You need to have a MAGE server started for it to work.

On Mon, 28 Apr 2014 14:24:32 +0300, Noxx Mage <[magenoxx@gmail.com](mailto:magenoxx@gmail.com)> wrote:

Hi,

I'm really interested in the progress done for the fx client.  
Am I right saying that once done such client can be run in browser?

One more question: is the following link the right place to get latest version?

<https://github.com/magefree/fx-client>

Best Regards,  
Noxx

2014-04-07 18:49 GMT+05:30 jdub1581 <[jdub1581@gmail.com](mailto:jdub1581@gmail.com)>:

Yeah, build time was horrible, removed sets, kept mage-commons though, and the build is now bearable.

I'm still working on the "HomePage" having a little creative block, as far as layout.. I wanna create a "Carousel" for the Menu's, ie: GameLobby, DeckEditor, CollectionViewer, Settings .. And create the components for Server Msgs, and Maybe add the Chat Panel there?

\*IF anyone wants to Contribute to Layout Ideas:\*create a FXML file and place a Rectangle with Label(description of what it is) where you think it should go..  
copy and paste Code here, or upload to My Project

Not sure how far you got on Properties (leaving that to you), but If you can create a set for GUI that'd be cool,  
I'd like, double-> layoutX & Y, double-> prefWidth & Height, String-> background-URL (for background Img)  
I remember before talking about making the client customizable and this is an incredibly easy feature to implement, Img can be local or Web based, By default will be set to the Ship img we have, with an Option under settings to set custom Img via "String".

As far as those Libraries go, JFXtras has for a long time tried to add components that are not available in JFX-core, same with controlsFX.

like JFXtras has a "CirclePane" which will layout child-nodes in a circle,  
Might be a cool component to have?

<http://jfxtras.org/> pics on site ...

Also ControlsFX has some as well, like Dialogs, and other UI things like  
an Info-Overlay (for cards maybe?)

controlsfx <<http://fxexperience.com/controlsfx/>> links on site ...



**North** <[robyter@gmail.com](mailto:robyter@gmail.com)> 28.04.14

an mage-developers

Here are the updates from me tinkering with JavaFX:

- There are some limitations regarding the view management. This means that this will need to be handled by us in such a way that we keep the nodes tree as light as possible. If there is anybody familiar with web development, this comes close enough to it but feels a bit more rigid and hard to work with. In the end I think it will end up being a nice client.
- Support for skinning is very easy to add. I propose to implement a minimalistic theme as the default. More fancy themes containing background images and what not can then be provided as zip archives. I haven't tested loading the full theme from a zip file but I believe it will work out of the box without any big issues.
- I was unpleasantly surprised that the only nodes that have selectable text are the TextField and TextArea. For me, those are inputs that should not be used for text output. Considering we only need this for the chat and the game log, I can ignore this limitation.

Here are some notes from what you will find in the repository:

- There is a dependency injection mechanism that I added to be able to have easy access to the server communication instances and to the settings.
- In order to respond to messages from the server, I added an implementation for MageClient that is an observable object. This one accepts handlers for all event types. To avoid memory leaks and the trouble of having to remove and event handler, the lists that store the handlers hold weak references. This means that the handlers themselves will be eligible for garbage collection even if they are present in the MageObservableServer. As a limitation, all handlers should be defined as instance fields in order to have a strong reference to them. Once the controller class gets "destroyed" the handlers will be automatically removed from the Observable server. I chose to do it like this because It would have been way too complicated to remove them otherwise.

I guess it is about time to start implementing the basic features to see how it behaves when creating more than a "hello world" chat application.



**jdub1581** <[jdub1581@gmail.com](mailto:jdub1581@gmail.com)> 28.04.14

an mage-developers

I'm still around and working on my end, my real job has been really busy lately, and the house I'm in just got put up for sale so things have been hectic. I do think about this everyday.

NORTH: go and check out DATAFX library, I've been checking it out and there are many things that may be very useful for you on the back end... ProcessChain(class) and many converters and concurrency utils..

<http://www.javafxdata.org/>

I hope to get some of the things I've been working on committed tonight after work...



**North** <[robyter@gmail.com](mailto:robyter@gmail.com)> 29.04.14

an mage-developers

Hi Jeff,

I looked at it and I think we are better off if we don't use it. Thank you for the pointer though.

[rant]

Now here is something related to JavaFX. I am so close to dismissing the civilized language when talking about this but the lack of nodes with selectable text is so limiting I can't even begin to describe it. In order to create the Game Log and the Chat Log functionality, I should write everything using HTML + JavaScript and then just execute JavaScript code on that WebView in order to update it. If you ask me this seems a bit retarded. I can easily do it but you tell me if this is the way it should be done. I'm almost thinking I should just build a web client and update the MAGE Server to simply be an HTTP server based on WebSockets.

One reason why we want to have a non web client is because of the graphic copy-written content.

[/rant]



**North <robyter@gmail.com>** 30.04.14

an mage-developers

I finished writing the LogPane component using the WebView. I am concerned with using such components because of the memory required to initialize a webkit instance for each instance of this component. Here is me hoping there won't be that many.

There is one thing that I did not properly do, namely make sure the web view extends to to fit the pane it is in. Jeff, do you have any tips regarding that?



**jdub1581 <jdub1581@gmail.com>** 30.04.14

an mage-developers

For binding a "View" to fit the size of a 'pane'  
couple of ways, on initialization can set prefWidth/height to be bound to 'Pane' w/h, Or use the constraints

like this: from javadoc

AnchorPane Example:

```
AnchorPane anchorPane = new AnchorPane();  
// List should stretch as anchorPane is resized  
ListView list = new ListView();  
AnchorPane.setTopAnchor(list, 10.0);  
AnchorPane.setLeftAnchor(list, 10.0);  
AnchorPane.setRightAnchor(list, 65.0);  
// Button will float on right edge  
Button button = new Button("Add");  
AnchorPane.setTopAnchor(button, 10.0);  
AnchorPane.setRightAnchor(button, 10.0);  
anchorPane.getChildren().addAll(list, button);
```

Or in FXML using the layout constraints, which is the same as above but done in scenebuilder





**jdub1581** <**jdub1581@gmail.com**> 30.04.14

an mage-developers

Regarding the selectable text issue, Not sure what you need it for but you can extend any node, and add a `StringProperty` and create the get and set text options.. but I'm sure you already considered that.



**robyter@gmail.com** 30.04.14

an mage-developers

Actually I didn't consider using a `StringProperty`. I quite a noob in terms of JavaFX stuff. could you send me a class that extends a `Text` node and is selectable. I'm quite interested in how that works.

All my google attempts to find a solution for this problem led me to either using a text input/area or `WebView`.



**jdub1581** <**jdub1581@gmail.com**> 01.05.14

an mage-developers

Well I'm still not quite sure what you're looking for but I think i get it... I saw your `LogPane` in the `ChatPanel`,

What I did in the original fx-client was used a `ListView` and created my own custom cell for rendering multiple labels for different parts of the msg...

create a manager for server updates and bam...

2 Anhänge

Vorschau für Anhang "MessageCellFactory.java" ansehen



Vorschau für Anhang "Message.java" ansehen



**jdub1581** <[jdub1581@gmail.com](mailto:jdub1581@gmail.com)> 01.05.14

an mage-developers

forgot to add these...

2 Anhänge

Vorschau für Anhang "ChatPanel.fxml" ansehen



Vorschau für Anhang "ChatPanel.java" ansehen



**robyter@gmail.com** 01.05.14

an mage-developers

I know about these. The previous version of the chat panel was build in a similar fashion.

Here is a message node: <https://github.com/magefree/fx-client/blob/1f9ea99b87aafcec04601dcfba44882e1c5a7769/src/main/java/mage/fxclient/chat/MessageNode.java>

Here is the controller using an observable list: <https://github.com/magefree/fx-client/blob/1f9ea99b87aafcec04601dcfba44882e1c5a7769/src/main/java/mage/fxclient/chat/ChatPanelController.java>

When I say selectable text I mean the ability to mark text, copy it to the clipboard and then paste it somewhere else. I guess I'll just add a tiny button somewhere that copies all the log to the clipboard.



**jdub1581** <[jdub1581@gmail.com](mailto:jdub1581@gmail.com)> 02.05.14

an mage-developers

OK, I gotcha now...

after some searching, possible solutions include:

- <http://stackoverflow.com/questions/17456716/how-to-highlight-a-row-if-it-contains-specific-text-in-javafx> (not exactly what you want but a good base)
- TextFlow class in jdk..
- [InputMethodEvent](#)
- [InputMethodHighlight](#)
- [InputMethodRequests](#)

Although, just in my opinion a ListView approach may be much better, due to the fact that it uses a virtual flow so that only visible cells are rendered keeping nodes to a minimum.



**North <robyter@gmail.com>** 03.05.14

an mage-developers

The WebView is only one JavaFX node which is an instance of webkit (browser engine). All the nodes live outside of the JavaFX context. The only concern is how resource heavy this node is from the start. Everything that happens in this node is from my point of view too light to care about.

Regarding the links you sent me, how should I put this? It makes me wonder if I know how to express myself.

In the end, all I can say is that what I desire is not implemented and is not easy or simple to implement. As a consequence, I'll either keep the web view implementation, or switch to a simple implementation that does not provide the option to select and copy (CTRL + C) the text.

As a side note, I looked a bit more in the source code of JavaFX... wow... what a mess. I don't know if I can say the same thing for Swing because I have not played enough with it but wow. The names they chose for some stuff makes my skin crawl.

- I mean, an Observable object becomes invalidated and the invalidated method gets called on the InvalidationListener?
- A control is Skinnable and I have no idea what a Skin really is even after I looked in the source code. All I know is that all the code is in the Skin class and it contains all the nodes even if the Skin is not a node.

I mean this are just some examples and maybe I shouldn't be digging in their source code, but man... does it suck or what? I have to admit though, if you stick with the "nodes" they provide, it shouldn't be too difficult to build an application.

I just wish they spent some more time deciding how to name things because they totally failed in this department.

And I'll end this with a nice quote:

There are two hard things in computer science: cache invalidation, naming things, and off-by-one errors.

Regards,  
Robert



**jdub1581 <jdub1581@gmail.com>** 04.05.14

an mage-developers

I was just throwing some ideas out there...

now Skinning on the other hand is a whole other can of worms... and what I've been playing with recently.

here is a good link to get started:

<http://www.guigarage.com/2012/11/custom-ui-controls-with-javafx-part-1/> finding part2 is a little weird but is in the first comment near bottom.



**Robert Biter <robyter@gmail.com>** 05.05.14

an Mage

Hey Jeff, I apologize for my snarky remark earlier. The suggestion on using the events was point on. I just felt it was way too much work. I guess I became too lazy.

This link you sent me about custom ui controls looks quite nice. Thank you.



**jdub1581 <jdub1581@gmail.com>** 05.05.14



an mage-developers

now you know why I've been looking at other libraries for controls... lol...  
Although I did just (kinda) create a Shelf control with a skin....

Behavior is still not a public api, but behavior only seems to handle events, which can be put in main Control class...

I did a whole lot of thinking regarding your LogPane ... I think I came up with a solution for it.

I propose using a TextFlow (add text nodes to it)  
add a mouse Pressed, and drag listener. ->  
On Pressed, create anchor points for a Rectangle , ->  
bind rectangle to mouse position, ->  
while Rectangle is active iterate through Flow.getChildren() ->  
if child.intersects(Rectangle.getBoundsInParent()) >> mark highLighted(change fill color maybe?) ->

At least it sounds good in theory...



**jdub1581** <[jdub1581@gmail.com](mailto:jdub1581@gmail.com)> 06.05.14

an mage-developers

Here is a simple one I threw together... very sloppy just proof of concept..

```
import javafx.application.Application;
import javafx.beans.property.DoubleProperty;
import javafx.beans.property.SimpleDoubleProperty;
import javafx.scene.Group;
import javafx.scene.Node;
import javafx.scene.Scene;
import javafx.scene.layout.StackPane;
import javafx.scene.paint.Color;
import javafx.scene.shape.Rectangle;
import javafx.scene.text.Text;
import javafx.scene.text.TextAlignment;
import javafx.scene.text.TextFlow;
import javafx.stage.Stage;
```

/\*\*

\*

\* @author Dub-Laptop

\*/

```
public class Test extends Application {
    private DoubleProperty rectInitX = new SimpleDoubleProperty();
    private DoubleProperty rectInitY = new SimpleDoubleProperty();
    private DoubleProperty rectX = new SimpleDoubleProperty();
    private DoubleProperty rectY = new SimpleDoubleProperty();

    private final Rectangle rect = createRect();

    @Override
    public void start(Stage primaryStage) {

        Group root = new Group();
        Text[] textNodes = new Text[]{
            new Text("A UI control is instantiated via its constructor. "),
            new Text("For example, a developer will call Button submitButton = new
Button(\"Submit\"). "),
            new Text("This will instantiate the button and run its constructor. "),
            new Text("As part of the constructor, it is suggested that a default CSS style class be
specified. "),
            new Text("By doing so this control will know what style class it belongs to, and it can
therefore be styled via CSS. "),
            new Text("For the case of Button, there is code inside the constructor that will do
something similar to getStyleclass().add(\"button\"). "),
            new Text("After the constructor is run, nothing else happens - no skin is instantiated,
no layout is run, and no CSS is run. "),
            new Text("The control will in fact have zero width and height. ")

        };

        for(Text t : textNodes){
            t.setWrappingWidth(250);
        }

        StackPane p = new StackPane();
        p.setPrefSize(800, 600);
        TextFlow flow = new TextFlow();

        flow.getChildren().addAll(textNodes);

        p.setOnMousePressed((e) ->{

            rect.setX(e.getX());
            rect.setY(e.getY());

            rectInitX.set(e.getX());
            rectInitY.set(e.getY());

            root.getChildren().add(rect);
```

```

});

p.setOnMouseDragged((e) -> {
    rectX.set(e.getX());
    rectY.set(e.getY());
    for(Node t : flow.getChildren()){
        if(((Text)t).getBoundsInParent().intersects(rect.getBoundsInLocal())){
            ((Text)t).setFill(Color.RED);
        }else{
            ((Text)t).setFill(Color.BLUE);
        }
    }
});

p.setOnMouseReleased((e) -> {
    // scan for nodes
    for(Node t : flow.getChildren()){
        if(((Text)t).getBoundsInParent().intersects(rect.getBoundsInLocal())){
            ((Text)t).setFill(Color.RED);
        }
    }
    root.getChildren().remove(rect);
});
p.getChildren().add(flow);
p.setStyle("-fx-background-color: rgba(0,0,0,0.25);");

rect.widthProperty().bind(rectX.subtract(rectInitX));
rect.heightProperty().bind(rectY.subtract(rectInitY));

root.getChildren().addAll(p);

Scene scene= new Scene(root, 1024, 700);

primaryStage.setTitle("Hello World!");
primaryStage.setScene(scene);
primaryStage.show();

}

/**
 * @param args the command line arguments
 */
public static void main(String[] args) {
    launch(args);
}

private Rectangle createRect(){
    Rectangle r = new Rectangle();
    r.setFill(Color.rgb(0, 0, 0, 0.5));

```

```

        return r;
    }

}

```



**jdub1581** <**jdub1581@gmail.com**> 09.05.14

an mage-developers

Ok, finally mad a commit, (forgot to re-label it tho) NOTE: There are several depends that are not used added to pom, they are just there for testing and possible components.

Not a whole lot in the way of progress, but some cool things were added, including:

BendingPages, a cool feature for the collection viewer, though som tweaking will be needed for multiple pages(maybe make a skin for TabPane, Pagination(?), or ListView. Took it from Ensemble 8 src.

A ShelfView control I made from the displayshelf example, still a work in progress, but functional, and was more of a test for creating skins.

\*ShelfView, will eventually extend ListView do decrease memory issues with creating too many nodes. So a new Skin for listview will be in the works.

Other than that just some project structuring..

I could really use some Ideas regarding Layout for the HomePage, ie:

having a main menu? or keeping the client similar to the current one with a menubar on top?

Should there be other panels there to display information such as Server messages?

Should we have a Profile section?

North: Also I need your brain for a good way to flip through the different screens, like DeckEditor, CollectionView, Lobby, etc... would your injection system be good for that? And also a good mechanism for loading cardInfo into the TableView, and a mechanism for Images...

On a side note in your ObservableServer, I'd like you to consider changing MageVersion to something like:

```

ObjectProperty<MageVersion> version = new SimpleObjectProperty<>(new
MageVersion(...));
@Override
public MageVersion getVersion() {
    return version.get();
}

```

This allows it to be bound later.. and a good habit in javafx as it's very property based (imho) although can become expensive if overused.

Also how do we want to implement Card, ie: basic view without images.



And another thing is Layout and interaction for BattleField... the basic stuff is self explanatory, but there are a few things I think could use stylistic improvements, like the whole panel on the right, I think that can just disappear all together. the chat area can be set to have transparency, and auto hide and show on msg received (without textInputArea), and on pressing (enter) showing it to send msg.

Also the player stats area need a major overhaul ... I've got a few Ideas in mind, but input is always nice...



**jdub1581** <jdub1581@gmail.com> 09.05.14

an mage-developers

Forgot to add... Should we use a PreLoader? should be a separate project if we do... Also this would be a good place to load default settings, loading the Server, doing card image update checks .. etc...



**robyter@gmail.com** 10.05.14

an mage-developers

I think we can add the preloader if the application starts slowly. I don't see things that will be intensive when starting up the client. All actions like checking images, connecting to the server ... should only be done by user request because they can be quite "heavy weight". Loading the settings is so fast it does not justify a preloader. In this scenario I'm going with let's add it when we need it. Getting everything else done is more important.



**robyter@gmail.com** 10.05.14

an mage-developers

This is quite nicely done. Should I even need to select text I'll use this approach. I think this is how the TextField works. Hmmm, I think i will get from there the necessary things and keep it really light weight.

But I will not implement it at this point. I think just adding a button to the LogPane that copies the text to the clipboard is more than enough.



**North <robyter@gmail.com>** 10.05.14

an mage-developers

The answer is in between:

On Friday, 9 May 2014 06:03:19 UTC+3, jdub1581 wrote:

I could really use some Ideas regarding Layout for the HomePage, ie:

having a main menu? or keeping the client similar to the current one with a menubar on top?

I guess we should keep the menu up top. People are used to it there.

Should there be other panels there to display information such as Server messages?

Should we have a Profile section?

I think we won't have trouble adding this things but I think we should focus on getting the minimum required to have a working client.

North: Also I need your brain for a good way to flip through the different screens, like DeckEditor, CollectionView, Lobby, etc... would your injection system be good for that? And also a good mechanism for loading cardInfo into the TableView, and a mechanism for Images...

The dependency injection can be used to provide a "ViewManager". This will be instantiated at application startup. In fact, a view factory sounds really good at this point and I'll introduce one when the code base will seem to need it. I'll add to my project a top menu and the navigation to the game lobby and another view that is light.

On a side note in your ObservableServer, I'd like you to consider changing MageVersion to something like:

```
ObjectProperty<MageVersion> version = new SimpleObjectProperty<>(new
MageVersion(...));
@Override
public MageVersion getVersion() {
    return version.get();
}
```

This allows it to be bound later.. and a good habit in javafx as it's very property based (imho) although can become expensive if overused.

I guess I should add the Property objects style in my arsenal if I'll work with JavaFX. The MageVersion won't change so it does not need to be an observable property. I think we'll need to change this to MageClientVersion and the server should be updated with a new version restriction only when a non backward compatible change is introduced in the communication layer. Also, the ObservableServer is UI agnostic so it should stay decoupled of JavaFX.

Also how do we want to implement Card, ie: basic view without images.  
This is a good question Considering the client will only work with "LightCard" objects, we'll need to decide how we represent them. We'll start off from what we receive from the server.

And another thing is Layout and interaction for Battlefield... the basic stuff is self explanatory, but there are a few things I think could use stylistic improvements, like the whole panel on the right, I think that can just disappear all together. the chat area can be set to have transparency, and auto hide and show on msg received (without textInputArea), and on pressing (enter) showing it to send msg.

Also the player stats area need a major overhaul ... I've got a few Ideas in mind, but input is always nice...

I like your ideas and how you styled the previous version of the FX client. I'm the kind of developer that implements it looking but ugly and then slaps on some CSS to make it look good. But I do work with the CSS and mind and provide a clean layout for it.



**jdub1581 <jdub1581@gmail.com>** 11.05.14

an mage-developers

So for Images, perhaps a Util class with static methods for getting image, perhaps a WeakObservableHashMap, if img isnt in map load it into map kind of deal? allowing unused images to get gc'd

When doing some research I came across this article: [javafx-tip-3-use-callback-interface](#). perhaps keeping this in mind will help with the ViewFactory.  
(perhaps working this into image loading? as the mediator for the list?)



**jdub1581 <jdub1581@gmail.com>** 11.05.14

an mage-developers

And for now, The Images I'll use in Layout, will be net based... to keep project size down... so Internet will be a must for now... Load time does not seem to be affected... and it does Look like they have updated css image calls quite nicely... from url's that is... was a bear before to get it right.

Still not sure how it will work with zip files... might have to har code like i was saying before... But I also think this would make Themes a lot easier being able to use net based resources... perhaps when the website is fixed, we can just load the base set of images to it, and call them from

there.. This would also keep image space to a bare minimum... maybe even for cards...??

My logic behind this is that for all practical purposes, playing would be across the internet anyways.. so a net connection shouldnt be a problem. And Images are already being downloaded from somewhere right?

With that in mind for those that wish, an option can be placed to download images as usual. Setting would include the option for "default image location" to be "Web/Net/Server/Whatever" or "User Defined"

@North

I'm going to add your InjectionProvider to my project so I can start building the GUI, thinking about scrapping the custom frame, and just set stage to be maximized, and non resizable.(MenuBar being the reasoning.) Also I think I can eliminate a lot of the Dialogs as well, By using the MenuBar DropLists like Settings, Connection, Symbols, and Images(options on right in current) The only thing I dont like is not being able to layout the menus as I want, ie: some on left some on right, some center. Maybe I'll look into toolbar and see about adding Single Menu MenuBars to it... And actually just tried as I write this and think this will be my approach.

I'm guessing that I should load the Global vars at start with Injection correct? (never really played with it before) just guessing based on your ObServer



**jdub1581 <jdub1581@gmail.com>** 17.05.14

an mage-developers

Important News: GA of SceneBuilder 2.0 is out!

<http://www.oracle.com/technetwork/java/javase/downloads/sb2download-2177776.html>



**jdub1581 <jdub1581@gmail.com>** 17.05.14

an mage-developers

@North: I pulled your project and saw your ViewFactory... I like it!, There is one issue I want you to consider, since you are fairly new to Javafx. the Parent class is imho inherently restrictive (maybe not for this project).

For example: There are no public getChildren() methods to manipulate child nodes, what is available is getChildrenUnmodifiable().

Sure you could use casting to get around this in most cases...

I personnaly when using fxml like to use the fx:root construct and extend

the root node in my controller class. Allowing me to use constructors and call methods.

Just food for thought...

I am in the process of implementing a RadialMenu instead of the MenuBar at top... I just hate have a full blank screen. It's almost done.. and pretty cool.

I think it will be a welcomed addition to the DeckEditor as well.



**robyter@gmail.com** 18.05.14

an mage-developers

I think using Parent is better. It makes it clear that one should only modify it from inside it's controller. The responsibility of the MainController is purely to navigate though the different views.

I did add a mechanism to change the view from outside the MainController but I didn't push the change because I noticed I didn't need it at that point. The point is that it is really simple. One creates a MessageBus class, added as a dependency to the injection provider and then just use publish/subscribe to do all the communication between views. I do not think we need it at this point, but when needed it is very simple to add.

I think I saw a radial menu component implemented in one of the libraries you mentioned. I agree with anything that enhances the user experience. I like to keep things as simple as possible. I think this is the secret for many things like good code or a good user experience.



**jdub1581 <jdub1581@gmail.com>** 25.05.14

an mage-developers

So North, I for the life of me cannot figure out how to get Injection working... I added your Provider to my proj, (dependencies are there too) Every time I try to add an Injection the Object keeps throwing npe's. I use it in the same way you do so I'm not sure whats going on, and google is not much help, other than saying add a beans.xml file... didn't work, nor did i see one in your project...

Also, I have been digging deeper into DataFX, especially the Flow api.. I think this is what I was looking for as far as a navigation standpoint.

It is rather simple to use, and reduces code immensely.

example: navigating from one class(view) to another is as simple as adding `@LinkAction(NewView.class)` to a button or other node or you can define links, actions, tasks, etc in the constructor of the Flow like:

```
... = new Flow(startView.class).withLinkAction(from.class, "name",
to.class).withAction(Controller.class, "actionName", FlowAction);
SubScene.setRoot(flow.start()); // uses StackPane as defaultViewContainer
```

other annotations are `@ActionMethod("name")` for methods, `@FXMLFlowAction("name")` for nodes(buttons or will handle on doubleclick of node) `@BackAction` on node returns to previous View..

I think this would work well for navigating the Main Pages.. and possibly some other areas

He also has tutorials up now at [GUIGarage](http://GUIGarage)

Also do you want to create a FX thread on slightlyMagic?

Still not a whole lot of progress as far a build goes, but I want to get the foundation done correctly, as I do not like to backpedal. So I will Implement a flow in my project for Main Navigation between things like DeckEditor and GameLobby, HomePage... Already have a basic one up and running.. Working on the GlobalLinks right now... It works quite seamlessly

What I am trying to acheive is creating a global NavigationBar for my Frame, I'd like to inject the Flow or FlowHandler into said navBar



**jdub1581 <jdub1581@gmail.com>** 25.05.14

an mage-developers

Ok I pushed what I've done so far ... take a look



**jdub1581 <jdub1581@gmail.com>** 27.05.14

an mage-developers

So I have done quite a bit the last couple days..

North: I added most of your classes to my proj.

My approach is using the Flow API from DataFX..

I cant seem to get your InjectionProvider to work for me even just copying all your code...  
So for "now" I am using datafx's ApplicationContext to register session and server to be called in controllers.

Everything so far works. Though I think I may break some FXML's into smaller components for ease of use in the future.  
here is a screen shot with your chatpanel:



**Robert Biter** <robyter@gmail.com> 27.05.14

an Mage

Here is how the InjectionProvider works. You add the dependencies to it. You can only have one "bean" for a class. I figured it was enough for our needs.

Take a look at the ViewFactory. This is where The injection is done. Basically it does more than just inject. It also creates an instance of that class using the default constructor. if needed, one can add a method to inject in existing instances too. If you are not instantiating your classes using the InjectionProvider than no injection is done.

Please let me know if this helped.

As far as DataFX goes, I am not a fan of it.