Building R-Shiny Applications

FAS6932 – Special Topics

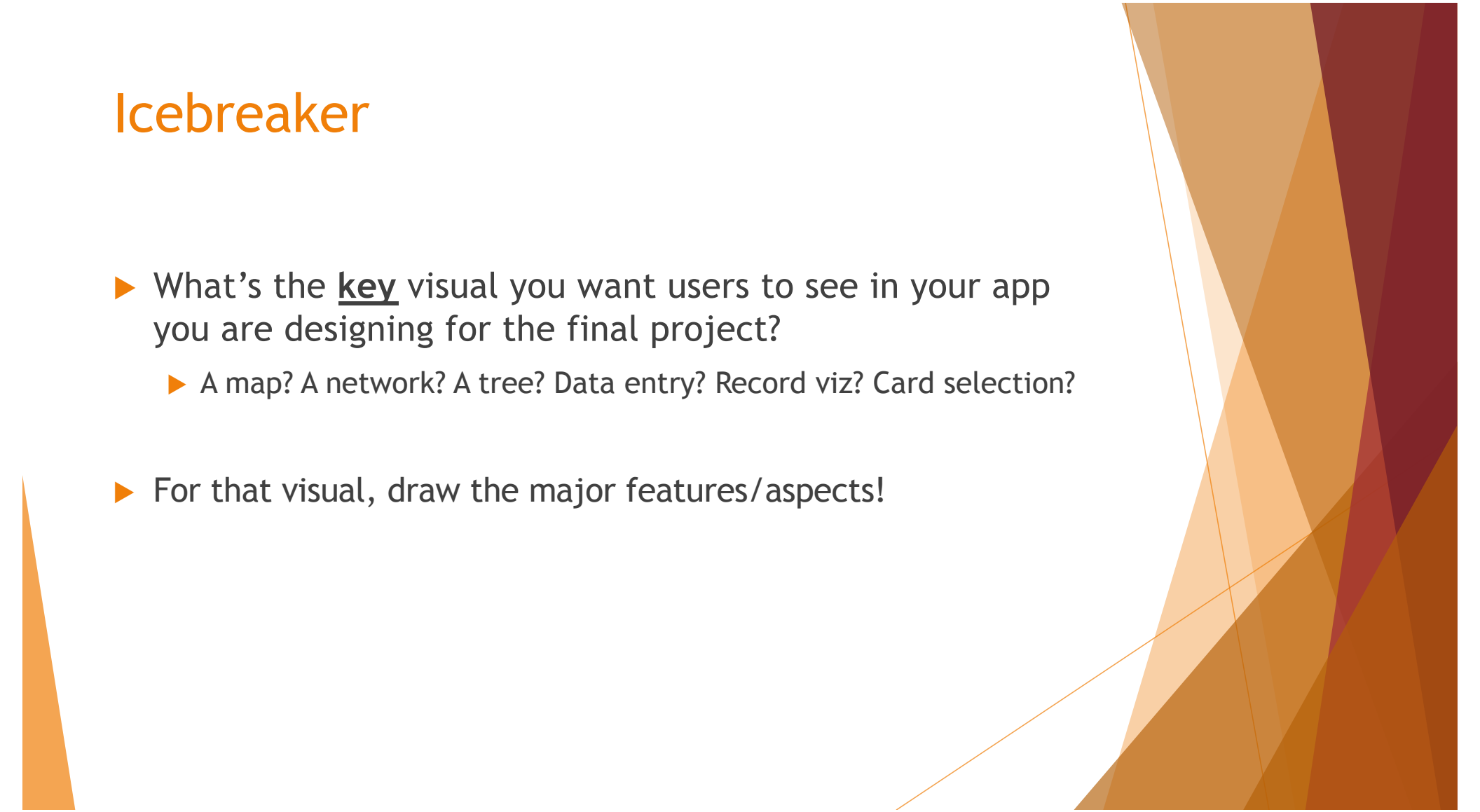
Summer C 2025

Day 4 – Advanced UI

- Morning Session
  - Rendering UI
  - Leaflet basics
- Lunch
- Afternoon Session
  - Leaflet basics +
  - Open design session
  - Hosting prep

Icebreaker

- What's the **key** visual you want users to see in your app you are designing for the final project?
  - A map? A network? A tree? Data entry? Record viz? Card selection?
- For that visual, draw the major features/aspects!



The backend of Shiny

- Ultimately, R-Shiny writes out HTML code

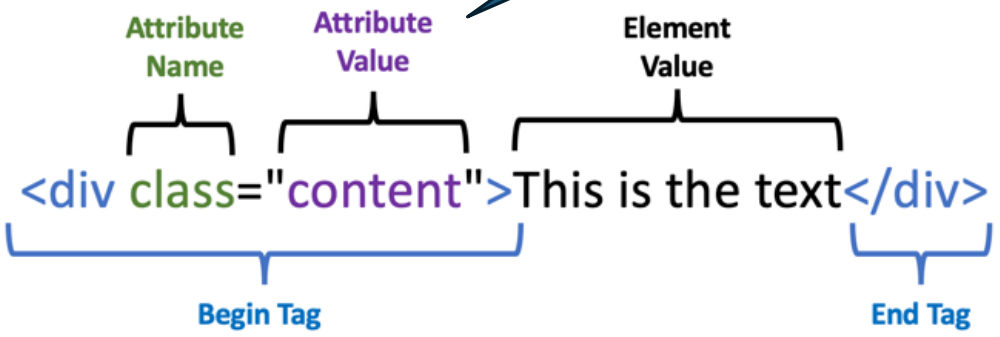
HTML uses tags

- You might be familiar with some of them:

Heading 1
Heading 2
Heading 3
Heading 4
Heading 5
Heading 6

```
<h1>Heading 1</h1>
<h2>Heading 2</h2>
<h3>Heading 3</h3>
<h4>Heading 4</h4>
<h5>Heading 5</h5>
<h6>Heading 6</h6>
```

We can modify the appearance of these elements with attributes



Custom Styling Sheets (CSS)

- CSS is a way to code the attributes of HTML tags globally
- Another way to say this is I want all <h1></h1> (Top header) tags to have these attributes
- In R-Shiny, we can write custom CSS to modify all the HTML pre-loaded tags
  - R-Shiny has some with built-in functions h1(…)
  - but all can be called with tags$h1(…)
- https://shiny.posit.co/r/articles/build/tag-glossary/

```
/* Styling the custom icon */
.custom_icon {
  background: url("custom_paw_icon.svg");
  background-size: contain;
  background-position: center;
  background-repeat: no-repeat;
  display: block;
}

```

How to style

- We can supply custom CSS in an external file
- We can supply custom CSS at the top of our R script
- We can supply HTML attributes to UI functions like
  - card, Widgets, labels
- We can wrap labels in HTML tags to call specific styles

The www folder

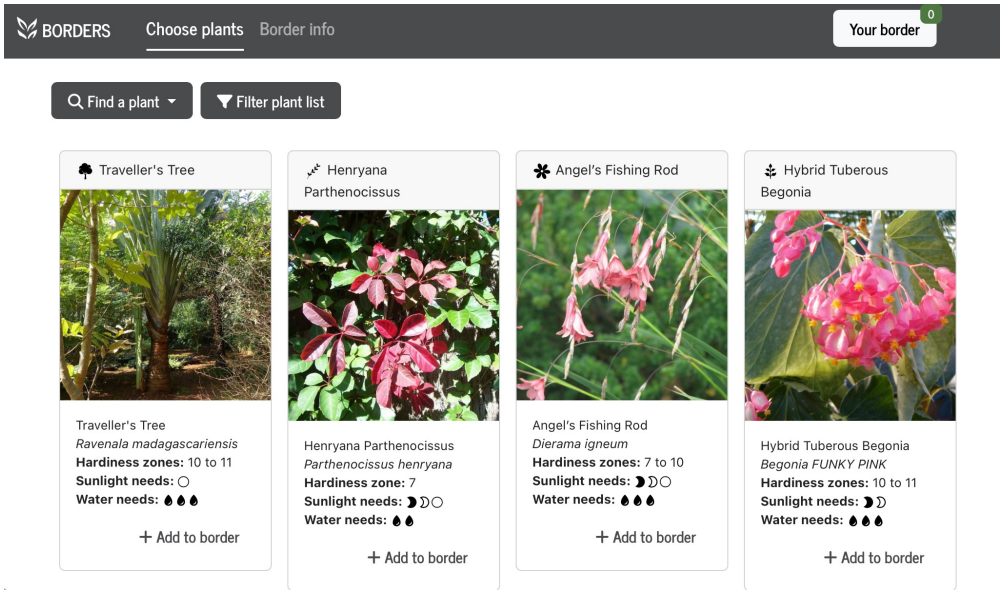
- Shiny (and HTML) reference this folder ‘www’ to grab all the bits that you might want to source on the page when web hosting
- This could be:
  - Images
  - Custom CSS
  - etc.
  - **NOT DATA**
- Confusingly, this does not render when running the app locally only when deployed on the web
  - so point to images with file = “./www/xxxx.png” when local
  - and point to images with url or src = “xxx.png” when web hosted

UI Resources

- R-Shiny tag glossary: https://shiny.posit.co/r/articles/build/tag-glossary/
- HTML formatting: https://www.w3schools.com/html/default.asp
- shinyWidgets: https://github.com/dreamRs/shinyWidgets
- Icons: ?icon
  - fontawesome: https://fontawesome.com/search?ic=free
  - glyphicons: https://getbootstrap.com/docs/3.3/components/
  - bsicons: ?bs_icon https://icons.getbootstrap.com
- Custom Fonts: https://fonts.google.com
- Dynamic Tabs: https://rstudio.github.io/bslib/reference/nav_select.html?q=nav_remove#ref-usage

rendering UI

- We can build the UI outside of the ui and server calls!
- So for apps like this:



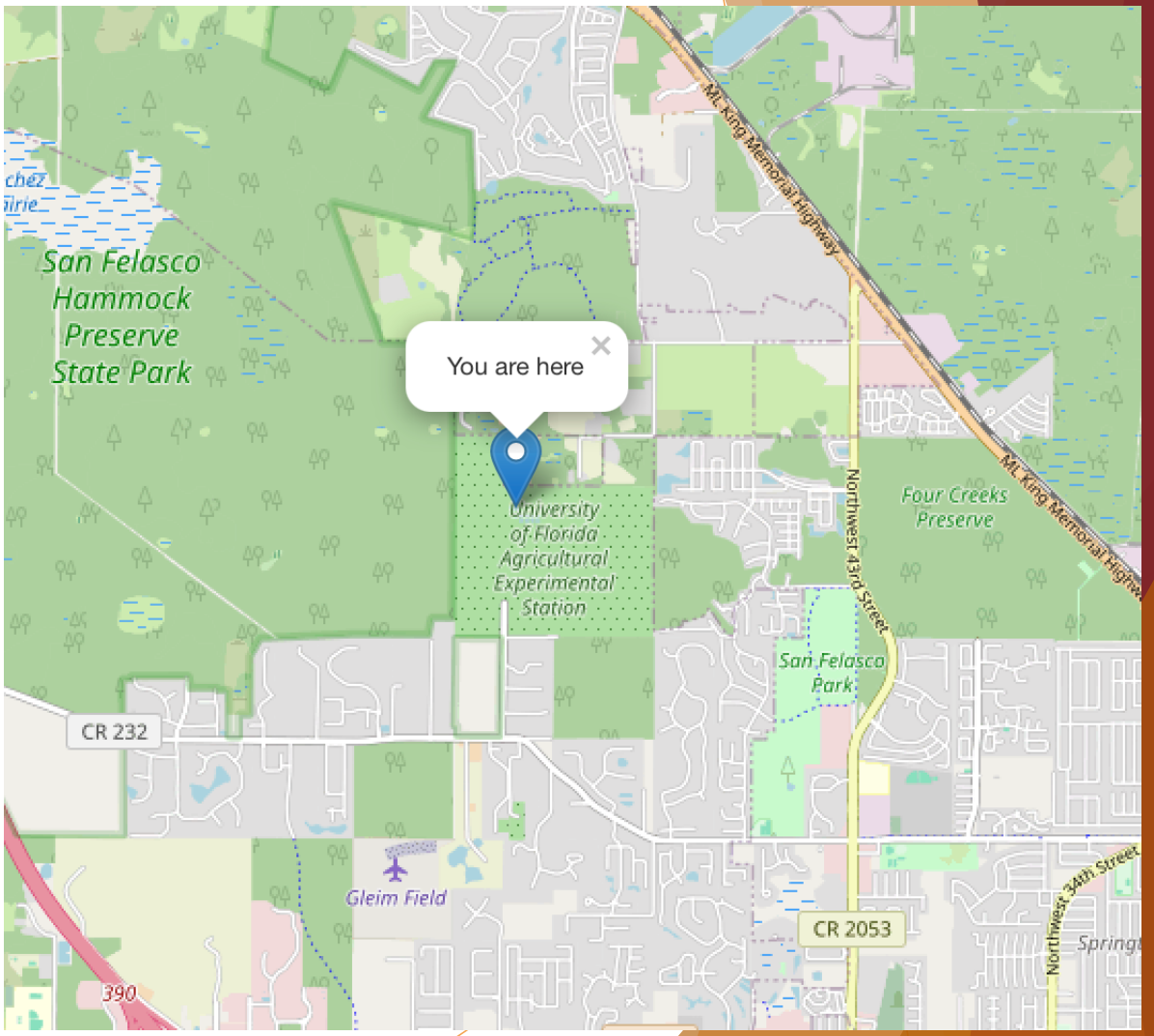
- We can write out all that UI ahead of time or even preload it!

Custom styling example

Run rendering UI/renderingUI_ex.R

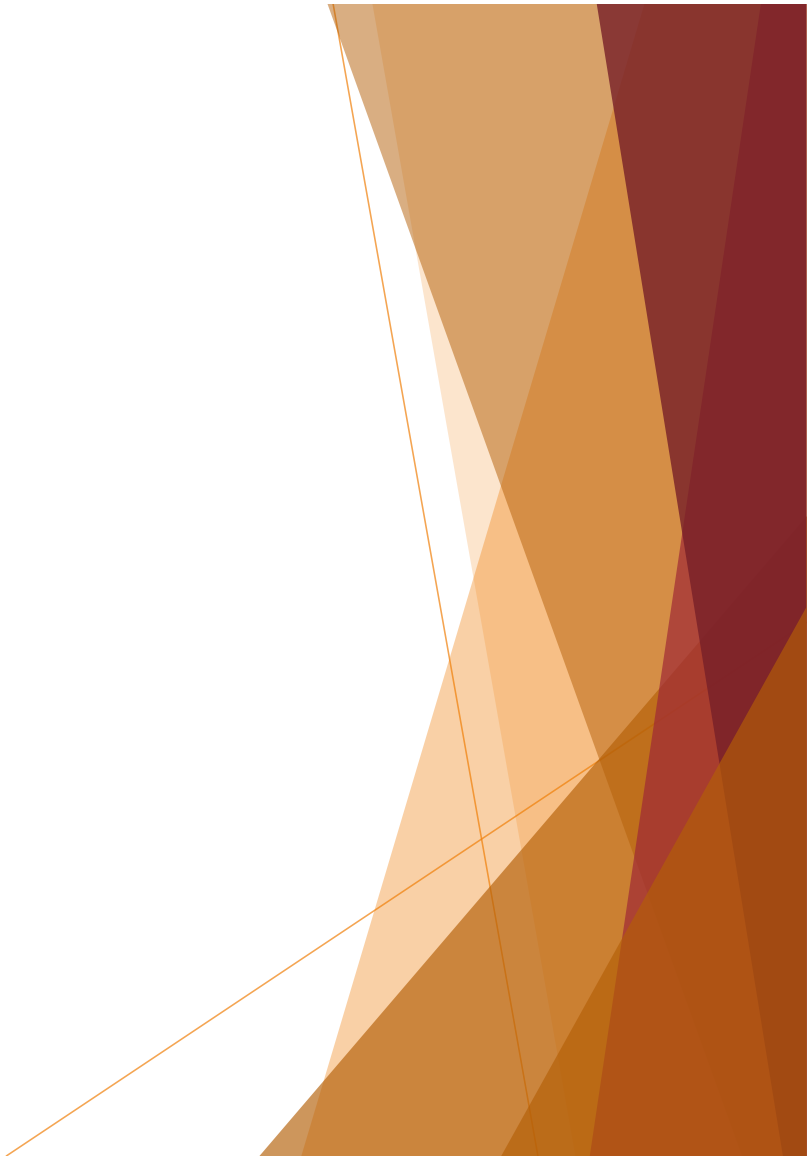
leaflet Basics

- https://rstudio.github.io/leaflet/articles/leaflet.html
- Leaflet is an open-source Javascript library
- leaflet is a R package that has R functions to access this library
- This results in a frustrating search experience for help as most will return JavaScript code, HTML code, or Python code (because Python also has a leaflet library, 😵)



leaflet basics

Run leaflet_basics.R



Leaflet resources

- Package Homepage: https://rstudio.github.io/leaflet/index.html
- Another leaflet tutorial: https://r-charts.com/spatial/interactive-maps-leaflet/
- leaflet extras providers: https://leaflet-extras.github.io/leaflet-providers/preview/
- leaflet extras 2: https://cloud.r-project.org/web/packages/leaflet.extras2/leaflet.extras2.pdf

leaflet more advanced

Run leaflet_ex/leaflet_ex.R