

Make a change

Mecklenburg!

MakeSigns logo will not print on final poster. For questions, call 800.347.2744 or email support@makesigns.com

©2018 MakeSigns

Template ID: pondering pack Size: 48x24

Computer Science '22 at Carleton College, WAVES Workshop at Michigan State University

Avida-ED...

- Is an educational software designed to help high school and undergraduate students learn about evolution.
- Uses the historical Avida program as its backend engine. Avida was developed first in 1993 by Dr. Charles Ofria.
- Since its initial conception, it has been incorporated into a number of undergraduate and high school science curriculums across the country.
- is actively being developed and maintained.



Introduction

Dragbars



Methodology



Results

What I Learned

Honestly, the way I think about web pages in general changed tremendously working on this project. I was constantly astounded by, and am still being astounded by the variety of ways JavaScript, and JQuery, and the dollar signs (who knew that \$ had so many meanings) and various Javascript libraries confuse each other! and give me frustration! but allow me to do magic on the page at the end :) And yes, something as simple as a dragbar, is rather complicated in its inner workings!

Steps I took to implement the Project:

1. I realized Avida-ED is a pretty big project (kudos to Diane!). I had to spend a few days simply looking at the code (literally) to get my eyes adjusted to all the namespaces and file structure!
2. I was tasked with creating the dragbars at first. I've never done that. Where should I start?
3. I made a proof of concept, sandbox type website to experiment with different components and mock up a dragbar.
4. I copied, pasted, modified my poc dragbar into the codebase, and tweaked a lot of things (it turned out) to make it actually compatible with the rest of the Avida-ED code and layout (Hello, maquettea grid!).

The Importance of Proof of Concept

What I learned is that, if you ever run into a situation where you don't know what the heck is going on in the codebase, and you don't know where to start? Do a proof of concept, it will save you big time. First, it is a great chance for you get yourself familiar with the Javascript functions that will be handy for your task. Second, it is a medium to communicate to stakeholders or your mentors what you are working on and how you plan on implementing the component onto the actual website. And lastly, and perhaps most importantly, it gives you a template, a kind of jumping off point to get you going (and with confidence) when you actually touch the codebase. Fear not!

Approach Code Like a Surgeon

What I mean by that is, my honest impression of working on the codebase was me constantly feeling like I was making small incisions in the code to insert whatever solution I had to offer the website (in this case, a dragbar). Like all beginner programmers, I was fearful. What if I break something really bad, and I mess something up, and the world comes to an end? (right...) Anyhow, I was always careful what I tweak and how I tweak it. I am a forgetful person, too. So I left a bunch of "yemi:(comment)" comments where I was changing anything, first leaving some breadcrumbs for me to pick back up if I do end up breaking something, and second to communicate to Diane what I was changing and why I was changing it, so if I do break something, she can at least follow what I was doing to perhaps offer up some higher level insight..(thank you, Diane!) But I learned that, if you approach code like a surgeon, and be very aware of every change you make precisely, you will save much energy later down the road where you need to back track not knowing where you messed up if you do (and you most likely will!)



Conclusion

1. Implement dragbars to allow for freely resizable windows.

2. Overhaul legacy dojo drag&drop and replace it with a new, more user-friendly
Dragula drag&drop framework.



Goals

Many thanks to Diane Blackwood, my advisor, without whom I would not have been able to achieve as much as I did. And to all of the WAVE mentors, colleagues, and Matthew!: Thank you so much for supporting me and encouraging me all this way; I really enjoyed hearing about everyone's contributions and progress throughout the workshop! It was inspiring, especially when coding wasn't going smooth, haha. So thank you. I hope our paths cross again in the future!



Acknowledgements

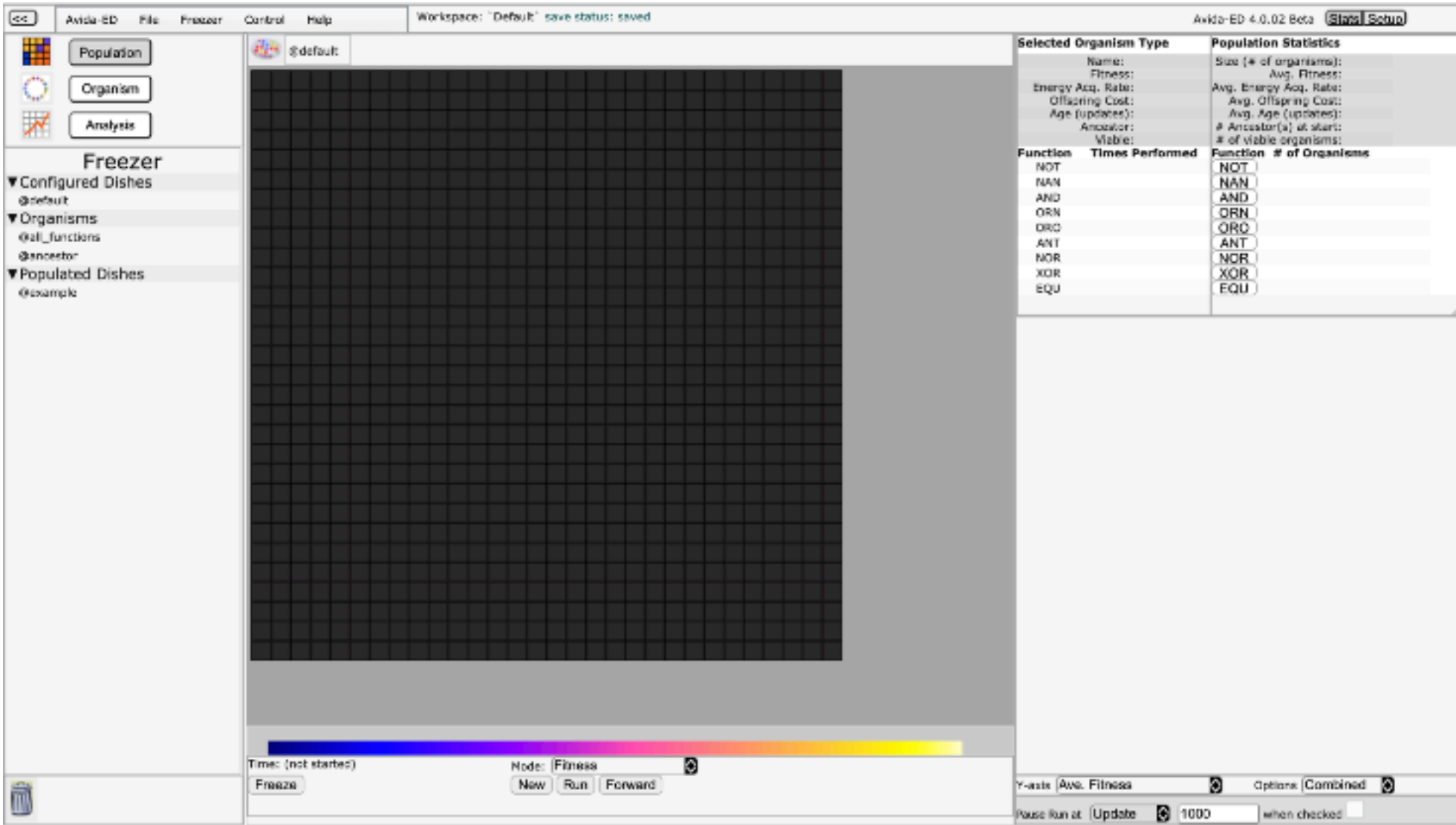
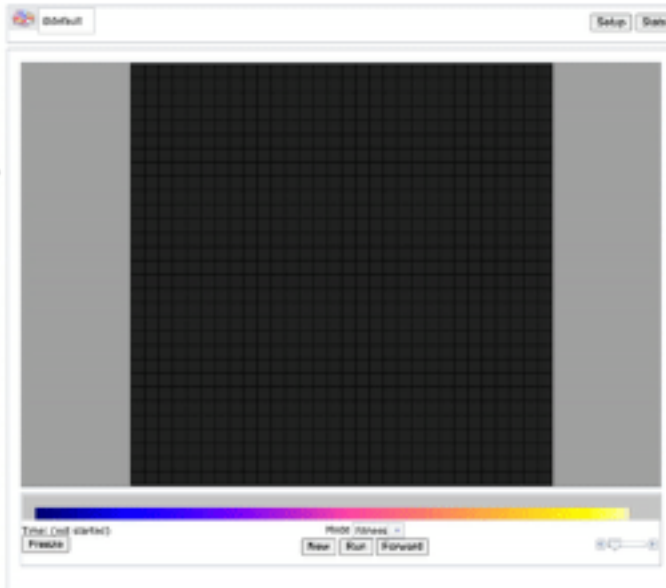


Fig 1. Original Avid-ED 4

One consideration was to maintain the look and feel of Avida-ED 3, the last stable release of Avida-ED. Avida-ED 3 had resizable windows.

Regulation
Organize
Analysis

Presets
→ Configured Presets
Default
→ Organisms
Sanitizer
Set Functions
→ Population States
Example



Selected Organism Type		Population Statistics	
Name:		Size (# of organisms):	
Fitness:		Avg. Fitness:	
Energy ACS, Ratio:		Max. Energy ACS, Ratio:	
Offspring Cost:		Avg. Offspring Cost:	
Age (generations):		Avg. Age (generations):	
Age (years):		# Ancestor(s) at start:	
Age (years):		# of males/organisms:	
Function	Fitness Performed	Function	# of Organisms
ACT		ACT	
ACT		ACT	
AND		AND	
AND		AND	
OR		OR	
OR		OR	
NOT		NOT	
NOT		NOT	
XOR		XOR	
XOR		XOR	

Y Axis: Average Fitness

Figure 2. Avidin-ED3

Dragula Drag & Drop

```
var population_colInfo = widthOfNav + "px 3px " + "auto 3px " + rightSideWidth;  
var organism_colInfo = widthOfNav + "px 3px " + "auto 3px " + rightSideWidth;  
var analysis_colInfo = widthOfNav + "px 3px auto";  
$('.all2lft').css("grid-template-columns", analysis_colInfo); /* yemi: you need to resize again  
$('.all3pop').css("grid-template-columns", population_colInfo);  
$('.all3org').css("grid-template-columns", organism_colInfo);
```



```
$(document).on('mousemove touchmove', function(e){  
  av.grd.drawGridSetupFn(); // yemi: redraw the grid  
  av.anl.AnaChartFn(); // yemi: redraw analysis grid  
  
  // yemi: need to account for both touch and mouse event  
  var x;  
  if(e.type == 'touchmove'){  
    var touch = e.originalEvent.touches[0] || e.originalEvent.changedTouches[0];  
    x = touch.pageX;  
  } else if (e.type == 'mousemove') {  
    x = e.pageX;  
  }  
}
```

```
var dra = dragula(containers, {
  isContainer: function (el) {
    return false; // only elements in drake.containers will be taken into account
  },
  moves: function (el, source, handle, sibling) {
    return true; // elements are always draggable by default
  },
  accepts: function (el, target, source, sibling) {
    if (target === source) {
      return true;
    }
    if ((source === av.dnd.ancestorBox) && (target === av.dnd.organIcon || target ===
      return true;
    }
    if (source === av.dnd.activeConfig && (target === av.dnd.fzConfig || target ===
      return true;
    }
  }
});
```

```
dra.on('drop', (el, target, source) => {  
  
  // el, target, source are dom objects aka stuff you could 'target.id' to  
  if ((target === av.dnd.activeConfig || target === av.dnd.ancestorBox) && av.grd.runState === 's  
    av.dom.newDishModalID.style.display = 'block'; // show the 'please save' modal  
    dra.cancel(); // cancel the drag event  
  } else if (target === av.dnd.activeConfig) {  
    av.dnd.landActiveConfig(el, target, source);  
  }  
})
```



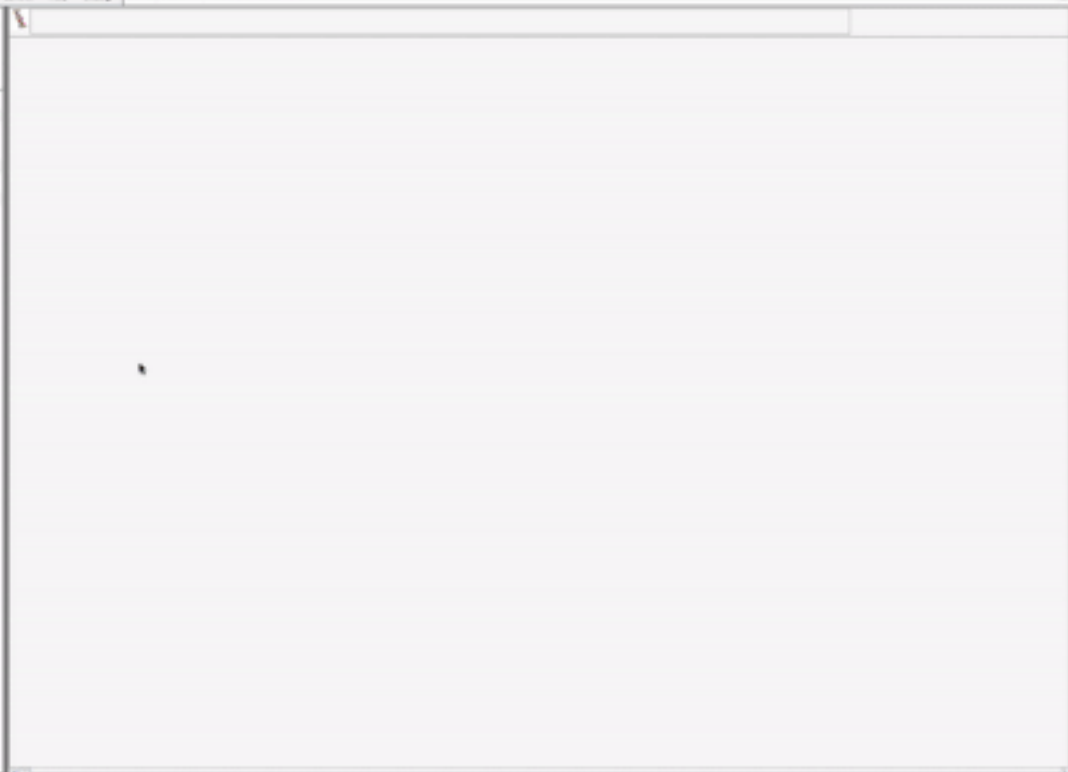

Figure 6. Drag & Drop in Avidia-ED 4 on Touch Device







- Freezer**
- Configured Dishes**
 - default
 - default2
 - 4hrs_fork_and_unfused
 - 4hrs_fork_and_unfused
 - Organisms**
 - cell_function
 - proteinase
 - Populated Dishes**
 - default





Avida-ED

File

Freezer

Control

Help

Debug

Workspace: `Default` save status: sav



Population



Organism



Analysis

Freezer

▼ Configured Dishes

@default
@default40sq
40sq_finite_grid_unlimited
30sq_finite_grid_unlimited

▼ Organisms

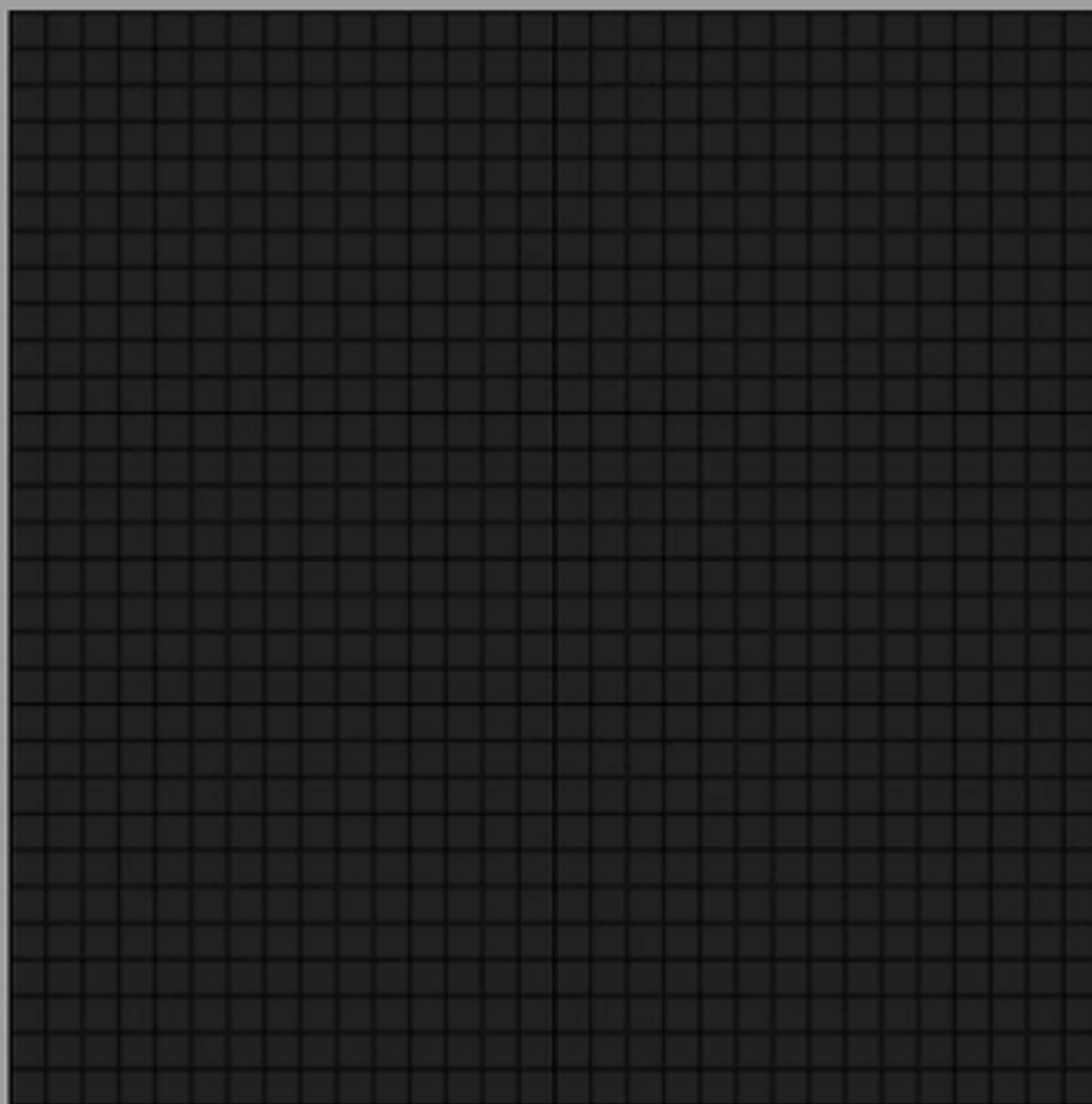
@all_functions
@ancestor

▼ Populated Dishes

@example



@default



Time: (not started)

Mode: Fitness



Freeze

New

Run

Forward

Selected Organism Type

Name:
Fitness:
Energy Acq. Rate:
Offspring Cost:
Age (updates):
Ancestor:
Viable:

Population Statistics

Size (# of organisms):
Avg. Fitness:
Avg. Energy Acq. Rate:
Avg. Offspring Cost:
Avg. Age (updates):
Ancestor(s) at start:
of viable organisms:

Function Times Performed

NOT
NAN
AND
ORN
ORO
ANT
NOR
XOR
EQU

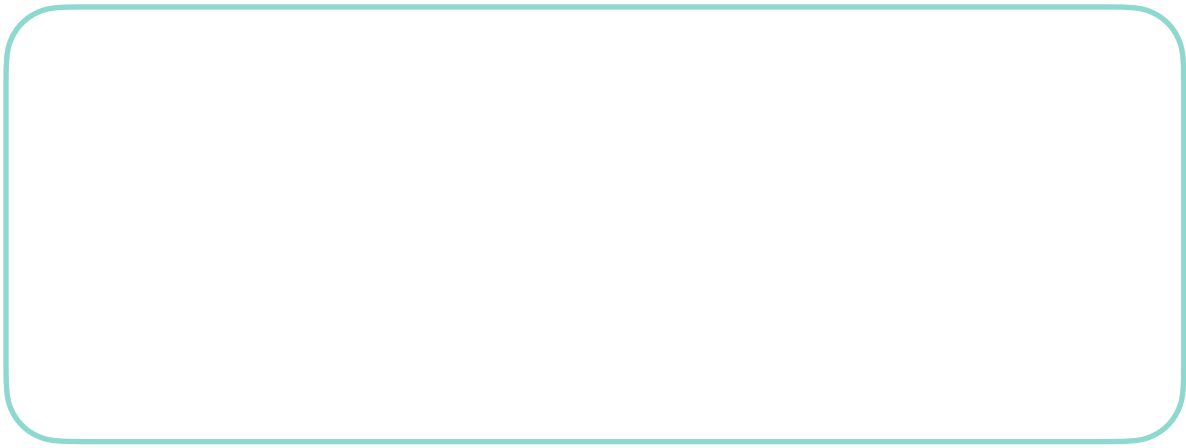
Function # of Organisms

NOT
NAN
AND
ORN
ORO
ANT
NOR
XOR
EQU



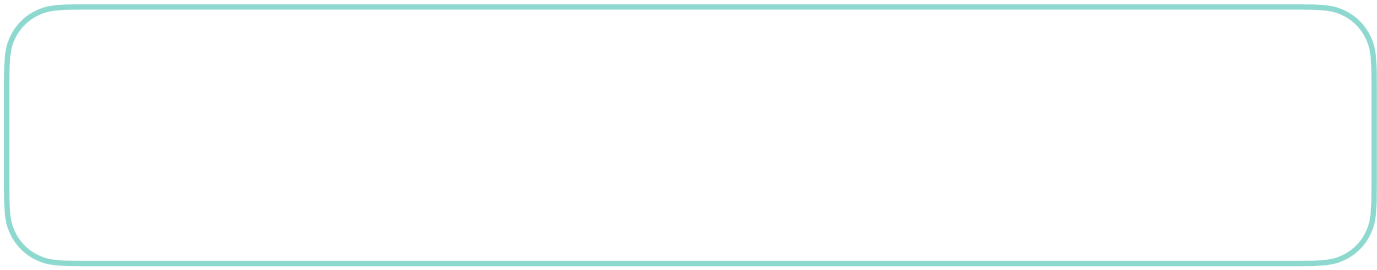
1. capture the mouse position

2. Figure out the layout grid column sizes




```
var rightSideWidth = $('#rightInfoHolder').css("width");  
var rightSideWidthNum = parseInt($('#rightInfoHolder').css("width")); /* yemi: extract only the  
var widthAvailable = window.innerWidth - rightSideWidthNum - 6; /* yemi: hard-coded 400px (right  
var percentage = (x / widthAvailable);  
var widthOfNav = widthAvailable * percentage;
```





3. change the cells of the grid



1. Set up the Dragula & Drop Engine



2. capture the 'drop' action


```
// when a configured dish is added to the config box
av.dnd.landActiveConfig = function (el, target, source) {
  'use strict';
  av.dnd.configFlag = 'normal';

  var ndx = -1;
  var klen = 0;
  var kk = 0;
  var str = '';

  var domid = el.id;

  // remove the existing configuration
  av.dnd.empty(target);
  av.dnd.insertToDOM(target, el);

  av.fzr.actConfig.actDomid = domid;
  av.fzr.actConfig.name = el.textContent;
  av.fzr.actConfig.fzDomid = source.id;
  av.fzr.actConfig.dir = av.fzr.dir[av.fzr.actConfig.actDomid];
}
```





3. Handles the 'drop' action





3a. Update the DOM

3.b. Update the backend message to communicate the change to Avida



Population

Organism

Analysis

Freezer

▼ Configured Dishes

@default

▼ Organisms

@ancestor

@all_functions

▼ Populated Dishes

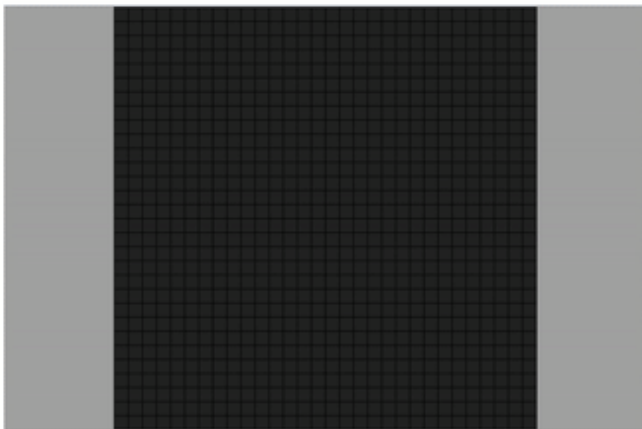
@example



@default

Setup

Stats



Time: (not started)

Freeze

Mode: Fitness

New

Run

Forward



Selected Organism Type

Name:

Fitness:

Energy Acq. Rate:

Offspring Cost:

Age (updates):

Ancestor:

Viable:

Function:

Times Performed

NOT

NAN

AND

ORN

ORD

ANT

NOR

XOR

EQU

Population Statistics

Size (# of organisms):

Avg. Fitness:

Avg. Energy Acq. Rate:

Avg. Offspring Cost:

Avg. Age (updates):

Ancestor(s) at start:

of viable organisms:

Function: # of Organisms

NOT

NAN

AND

ORN

ORD


ANT

NOR

XOR

EQU

Y-axis: Average Fitness

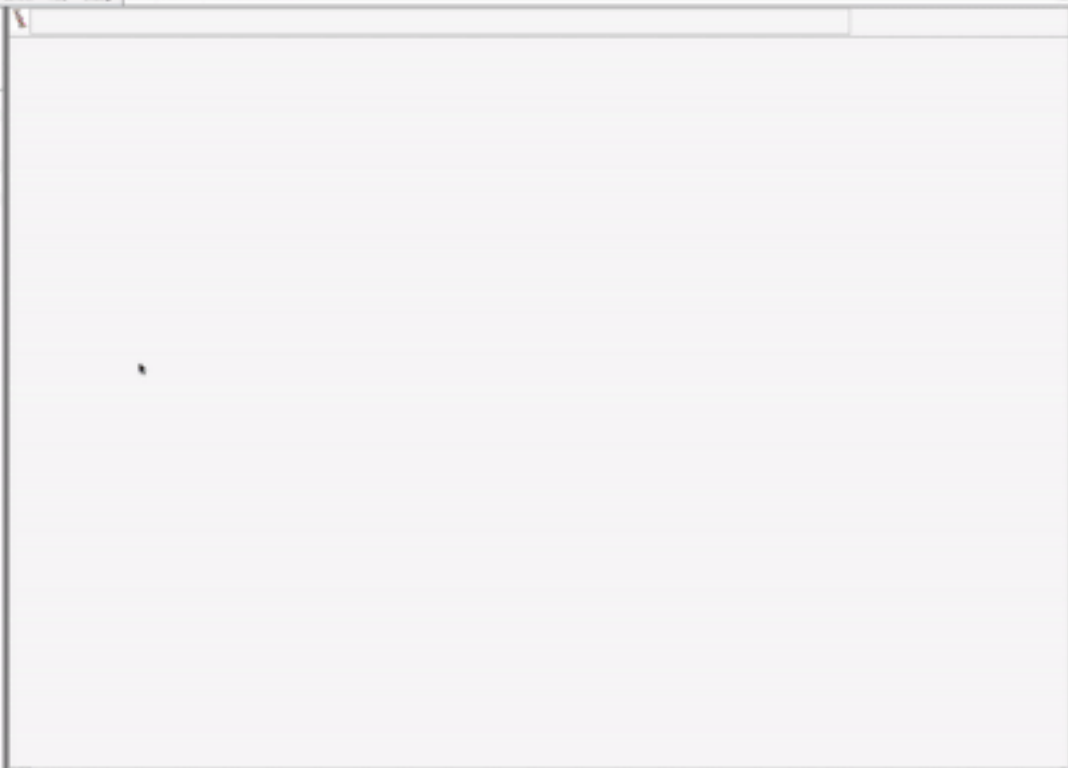


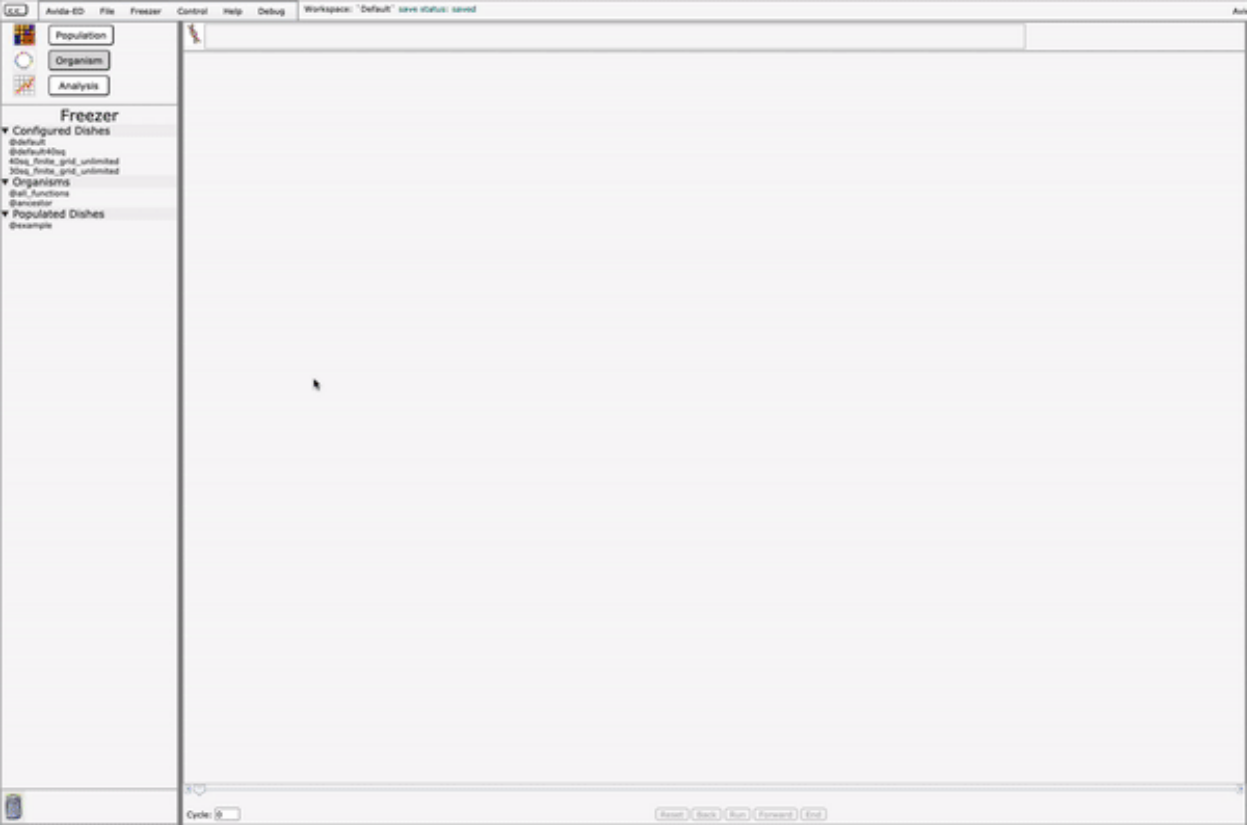
Population

Organism

Analysis

- Freezer**
- Configured Dishes**
 - default
 - default0000
 - 0000_0000_0000_0000
 - 0000_0000_0000_0000
 - Organisms**
 - 0000_000000
 - 00000000
 - Populated Dishes**
 - 00000000







Avida-ED

File

Freezer

Control

Help

Debug

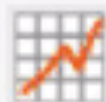
Workspace: `Default` save status: sav



Population



Organism



Analysis

Freezer

▼ Configured Dishes

@default
@default40sq
40sq_finite_grid_unlimited
30sq_finite_grid_unlimited

▼ Organisms

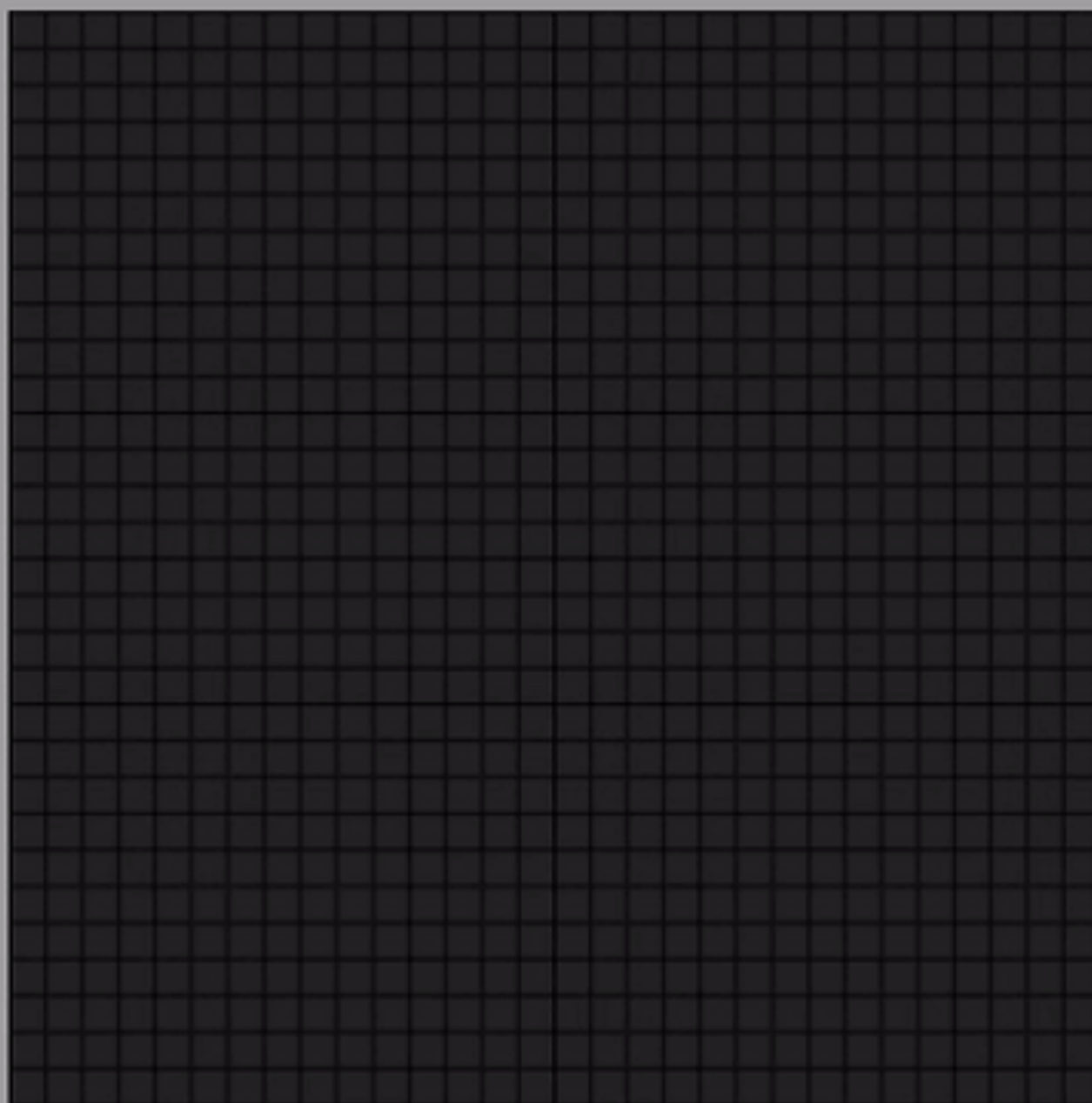
@all_functions
@ancestor

▼ Populated Dishes

@example



@default



Time: (not started)

Mode: Fitness



Freeze

New

Run

Forward

Selected Organism Type

Name:
Fitness:
Energy Acq. Rate:
Offspring Cost:
Age (updates):
Ancestor:
Viable:

Population Statistics

Size (# of organisms):
Avg. Fitness:
Avg. Energy Acq. Rate:
Avg. Offspring Cost:
Avg. Age (updates):
Ancestor(s) at start:
of viable organisms:

Function Times Performed

NOT
NAN
AND
ORR
ORR
AND
NOR
XOR
EQU

Function # of Organisms

NOT
NAN
AND
ORR
ORR
AND
NOR
XOR
EQU

Selected Organism Type

Name:
Fitness:
Energy Acq. Rate:
Offspring Cost:
Age (updates):
Ancestor:
Viable:

Population Statistics

Size (# of organisms):
Avg. Fitness:
Avg. Energy Acq. Rate:
Avg. Offspring Cost:
Avg. Age (updates):
Ancestor(s) at start:
of viable organisms:

Function Times Performed

NOT
NAN
AND
ORN
ORO
ANT
NOR
XOR
EQU

Function # of Organisms

NOT
NAN
AND
ORN
ORO
ANT
NOR
XOR
EQU



Avida-ED 4: Usability Improvements

Yemi Shin

Computer Science '22 at Carleton College, WAVES Workshop at Michigan State University

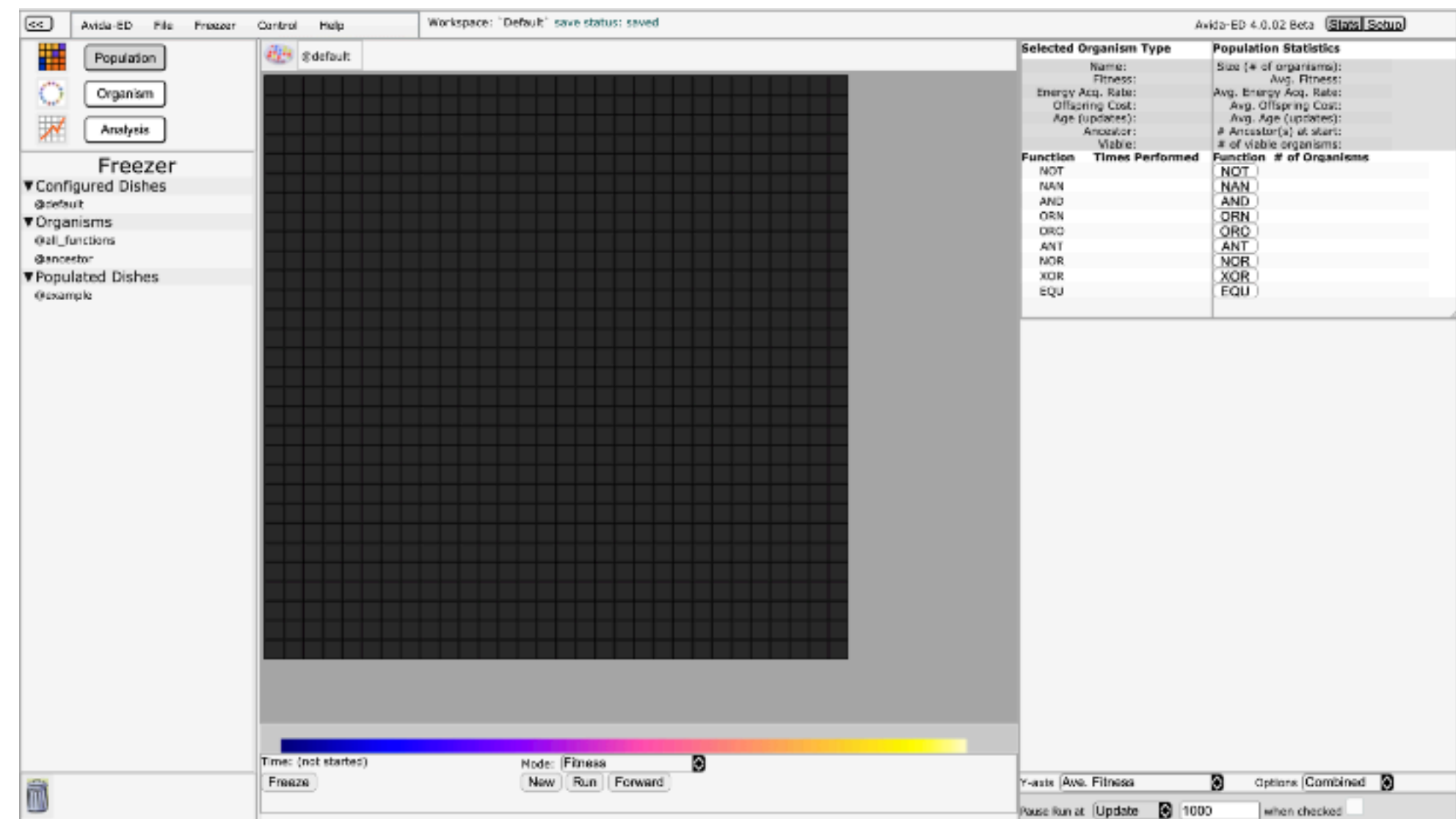
Introduction

Avida-ED...

- Is an educational software designed to help high school and undergraduate students learn about evolution.
- Uses the historical Avida program as its backend engine. Avida was developed first in 1993 by Dr. Charles Ofria.
- Since its initial conception, it has been incorporated into a number of undergraduate and high school science curriculums across the country.
- is actively being developed and maintained.

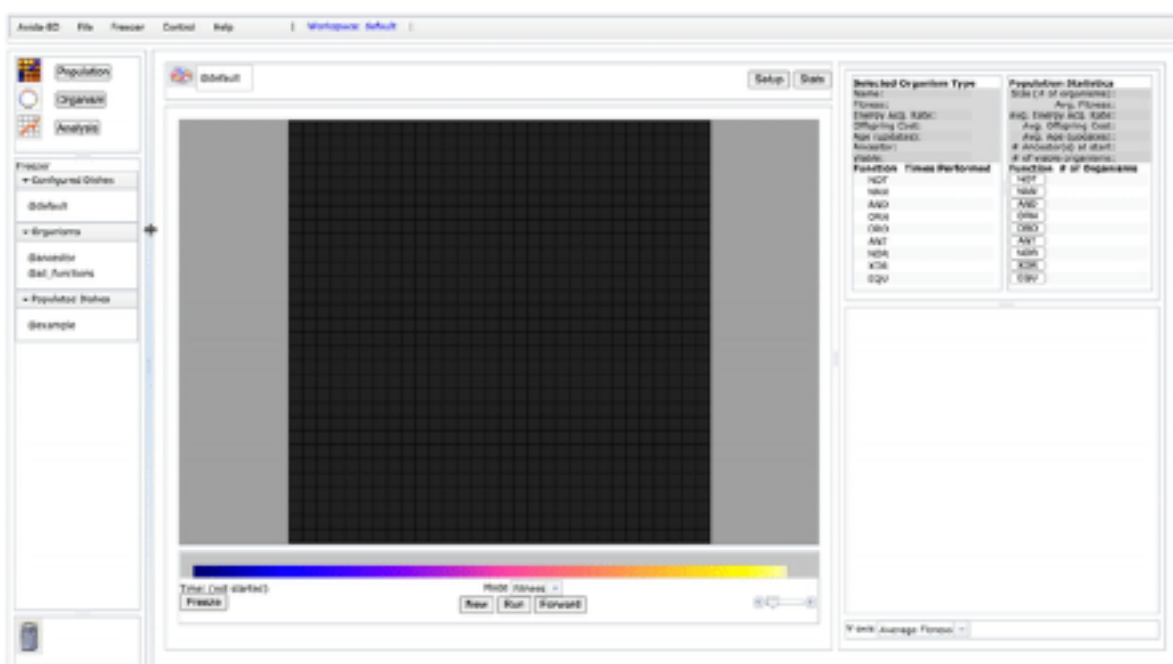
Goals

- Implement dragbars to allow for freely resizable windows.
- Overhaul legacy dojo drag&drop and replace it with a new, more user-friendly Dragula drag&drop framework.



<Figure 1. Original Avida-ED 4>

One consideration was to maintain the look and feel of Avida-ED 3, the last stable release of Avida-ED. Avida-ED 3 had resizable windows.



<Figure 2. Avida-ED 3>

Methodology

Dragbars

reSizePageParts.js

```
$(document).on('mousemove touchmove', function(e){
  av.grd.drawGridSetupFn(); // yemi: redraw the grid
  av.anl.AnaChartFn(); // yemi: redraw analysis grid

  // yemi: need to account for both touch and mouse event
  var x;
  if(e.type == 'touchmove'){
    var touch = e.originalEvent.touches[0] || e.originalEvent.changedTouches[0];
    x = touch.pageX;
  } else if (e.type == 'mousemove') {
    x = e.pageX;
  }
```

1. Capture the mouse position

```
var rightSideWidth = $('#rightInfoHolder').css('width');
var rightSideWidthNum = parseInt($('#rightInfoHolder').css('width')); // yemi: extract only the
var widthAvailable = window.innerWidth - rightSideWidthNum - 6; // yemi: hard-coded 400px (right
var percentage = (x / widthAvailable);
var widthOfNav = widthAvailable * percentage;
```

```
var population_colInfo = widthOfNav * "px 3px " + "auto 3px " + rightSideWidth;
var organism_colInfo = widthOfNav * "px 3px " + "auto 3px " + rightSideWidth;
var analysis_colInfo = widthOfNav * "px 3px auto";
$('.all2ift').css("grid-template-columns", analysis_colInfo); // yemi: you need to resize again
$('.all3pop').css("grid-template-columns", population_colInfo);
$('.all3org').css("grid-template-columns", organism_colInfo);
```

3. Change the css of the grid

Dragula Drag & Drop

dragulaDnd.js

```
var dra = dragula(containers, {
  isContainer: function (el) {
    return false; // only elements in drake.containers will be taken into account
  },
  moves: function (el, source, handle, sibling) {
    return true; // elements are always draggable by default
  },
  accepts: function (el, target, source, sibling) {
    if (target === source) {
      return true;
    }
    if (source === av.dnd.ancestorBox) && (target === av.dnd.organismIcon || target ===
    }
    if (source === av.dnd.activeConfig && (target === av.dnd.fzConfig || target ===
    }
    return true;
  }
});
```

dra.on('drop', (el, target, source) => {

```
// el, target, source are dom objects aka stuff you could 'target.id' to
if ((target === av.dnd.activeConfig || target === av.dnd.ancestorBox) && av.grd.runState === 's
av.dom.newDishModalID.style.display = 'block'; // show the 'please save' modal
dra.cancel(); // cancel the drag event
} else if (target === av.dnd.activeConfig) {
  av.dnd.landActiveConfig(el, target, source);
}
```

2. Capture the 'drop' action

```
// when a configured dish is added to the config box
av.dnd.landActiveConfig = function (el, target, source) {
  'use strict';
  av.dnd.configFlag = 'normal';

  var ndx = -1;
  var klen = 0;
  var kk = 0;
  var str = '';

  var domid = el.id;

  // remove the existing configuration
  av.dnd.empty(target);
  av.dnd.insertToDOM(target, el);

  av.fzr.actConfig.actDomid = domid;
  av.fzr.actConfig.name = el.textContent;
  av.fzr.actConfig.fzDomid = source.id;
  av.fzr.actConfig.dir = av.fzr.dir[av.fzr.actConfig.actDomid];
```

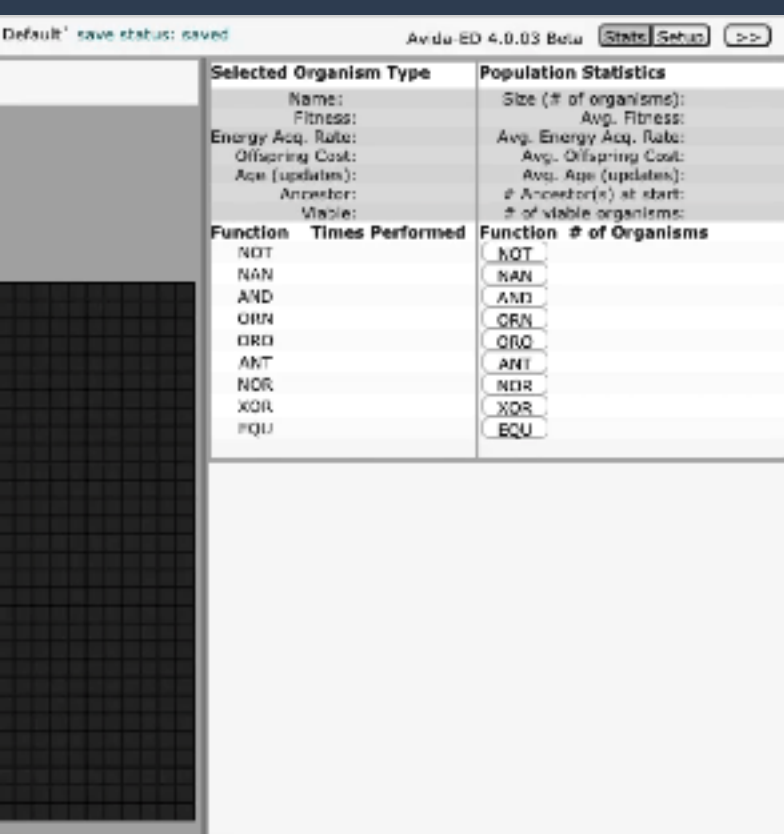
3. Handle the 'drop' action

3.a. Update the DOM

3.b. Update the backend message to communicate the change to Avida

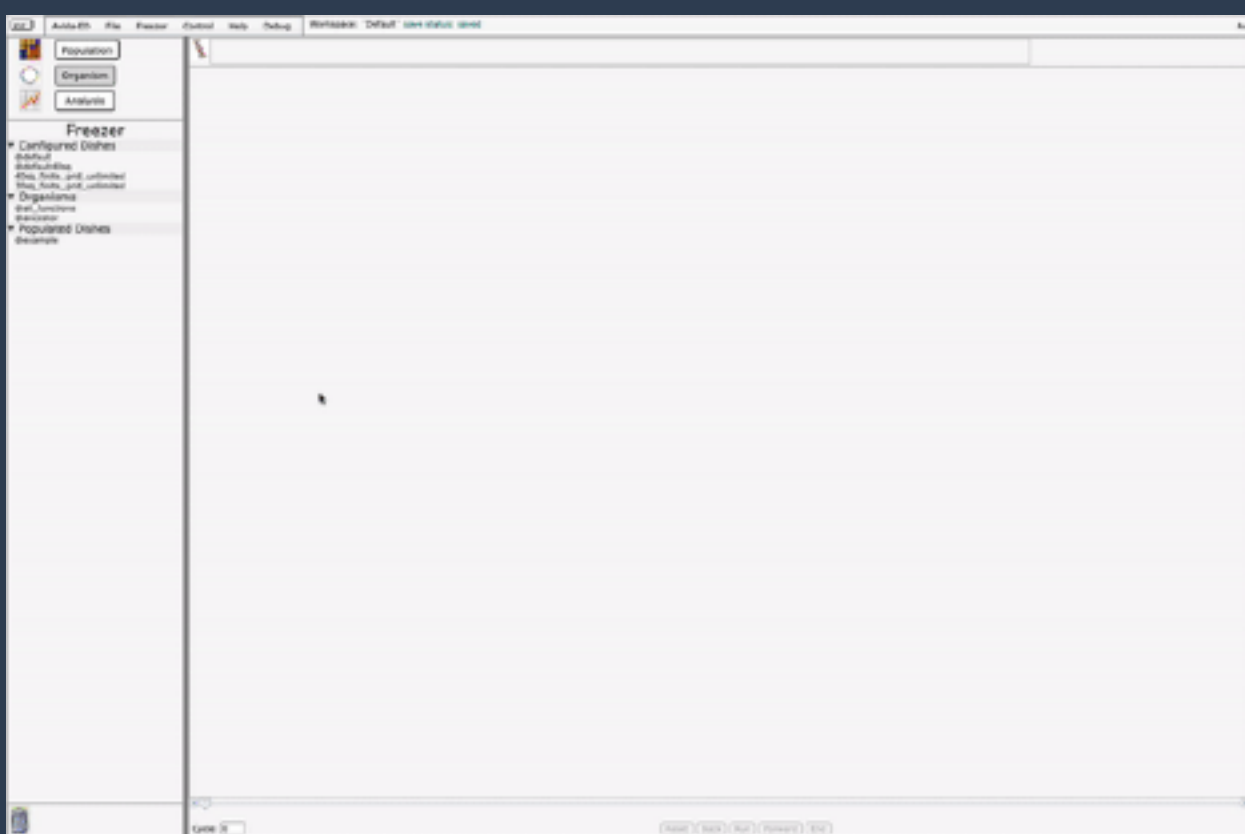
Results

Dragbars

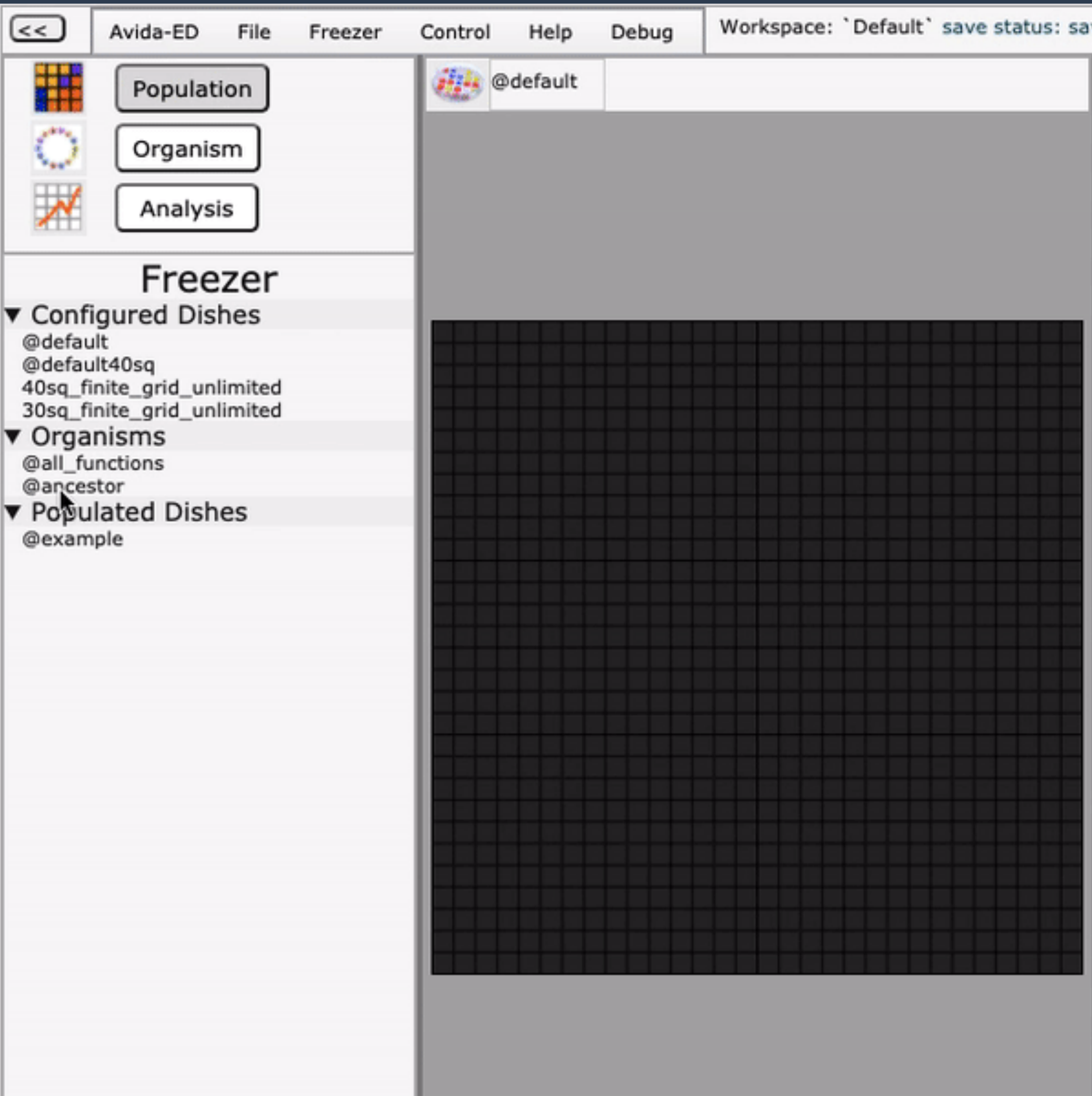


<Figure 3. Dragbars in Avida-ED 4 on Desktop>

Dragula Drag & Drop



<Figure 4. Drag&Drop in Avida-ED 4 on Desktop>



<Figure 5. Drag&Drop in Avida-ED 4 on Touch Screen>

Conclusion

What I Learned

Honestly, the way I think about web pages in general changed tremendously working on this project. I was constantly astounded by, and am still being astounded by the variety of ways JavaScript, and JQuery, and the dollar signs (who knew that \$ had so many meanings) and various Javascript libraries confuse each other! and give me frustration! but allow me to do magic on the page at the end :) And yes, something as simple as a dragbar, is rather complicated in its inner workings!

Steps I took to implement the Project:

- I realized Avida-ED is a pretty big project (kudos to Diane!). I had to spend a few days simply looking at the code (literally) to get my eyes adjusted to all the namespaces and file structure!
- I was tasked with creating the dragbars at first. I've never done that. Where should I start?
- I made a proof of concept, sandbox type website to experiment with different components and mock up a dragbar.
- I copied, pasted, modified my poc dragbar into the codebase, and tweaked a lot of things (it turned out) to make it actually compatible with the rest of the Avida-ED code and layout (Hello, maquette grid!).

The Importance of Proof of Concept

What I learned is that, if you ever run into a situation where you don't know what the heck is going on in the codebase, and you don't know where to start? Do a proof of concept, it will save you big time. First, it is a great chance for you get yourself familiar with the Javascript functions that will be handy for your task. Second, it is a medium to communicate to stakeholders or your mentors what you are working on and how you plan on implementing the component onto the actual website. And lastly, and perhaps most importantly, it gives you a template, a kind of jumping off point to get you going (and with confidence) when you actually touch the codebase. Fear not!

Approach Code Like a Surgeon

What I mean by that is, my honest impression of working on the codebase was me constantly feeling like I was making small incisions in the code to insert whatever solution I had to offer the website (in this case, a dragbar). Like all beginner programmers, I was fearful. What if I break something really bad, and I mess something up, and the world comes to an end? (right...) Anyhow, I was always careful what I tweak and how I tweak it. I am a forgetful person, too. So I left a bunch of "yemi:(comment)" comments where I was changing anything, first leaving some breadcrumbs for me to pick back up if I do end up breaking something, and second to communicate to Diane what I was changing and why I was changing it, so if I do break something, she can at least follow what I was doing to perhaps offer up some higher level insight..(thank you, Diane!) But I learned that, if you approach code like a surgeon, and be very aware of every change you make precisely, you will save much energy later down the road where you need to back track not knowing where you messed up if you do (and you most likely will!)

Acknowledgements

Many thanks to Diane Blackwood, my advisor, without whom I would not have been able to achieve as much as I did. And to all of the WAVE mentors, colleagues, and Matthew!: Thank you so much for supporting me and encouraging me all this way; I really enjoyed hearing about everyone's contributions and progress throughout the workshop! It was inspiring, especially when coding wasn't going smooth, haha. So thank you. I hope our paths cross again in the future!

