
MakeSigns

MakeSigns

Template ID: ponderingpeacock 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

- I was constantly astounded by, and am still being astounded by the variety of ways that JavaScript, and JQuery, and the dollar signs (who knew that $ had so many meanings) work together but at times are frustratingly incompatible with each other. They were the secret sauce to the magic on the screen, in the end, though.

- Something as simple as a dragbar or drag&drop, is rather complicated in its inner workings!

Steps I took to implement the Project:

1. 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 that make up a drawbar, and mocked up a dragbar apart from the main codebase.

4. I copied, pasted, modified my poc dragbar into the codebase, and tweaked several things to make it actually compatible with the rest of the Avida-ED code and layout (Hello, maquetta grid!)

The Importance of Proof of Concept

- It is a great chance for you get yourself familiar with the Javascript functions that will be handy for your task.

- 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 software.

- It gives you a template, a kind of jumping off point to get you going (and with confidence!) when you finally touch the codebase. Fear not!

Approach Code Like a Surgeon

If you approach code like a surgeon, and are very aware of every change you make precisely, you will save much time and energy later down the road when you might need to back track to locate the origin of some error blind sighted! (and you most likely will!)

Conclusion

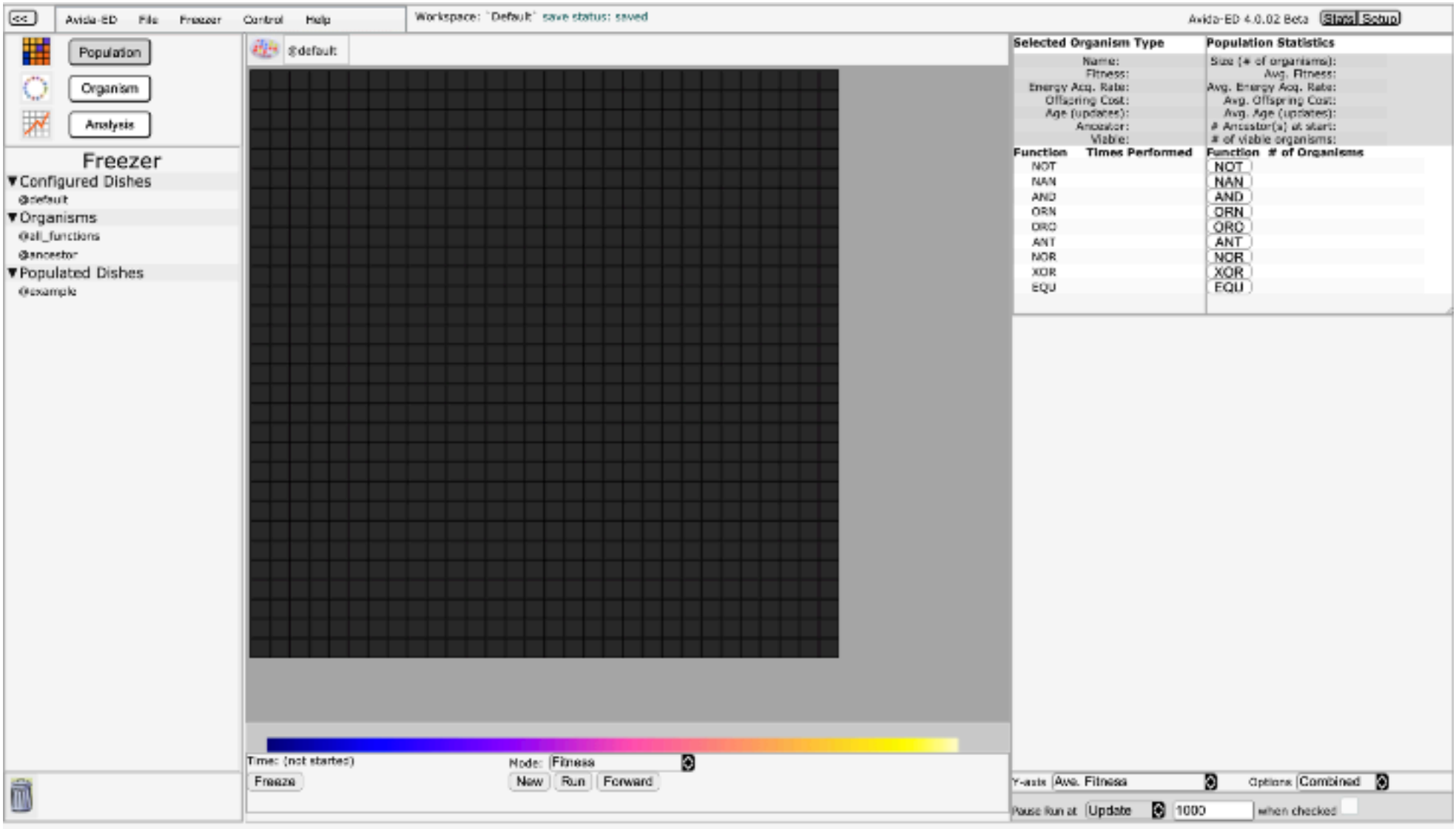
1.Implement dragbars to allow for freely resizable windows.

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

Goals

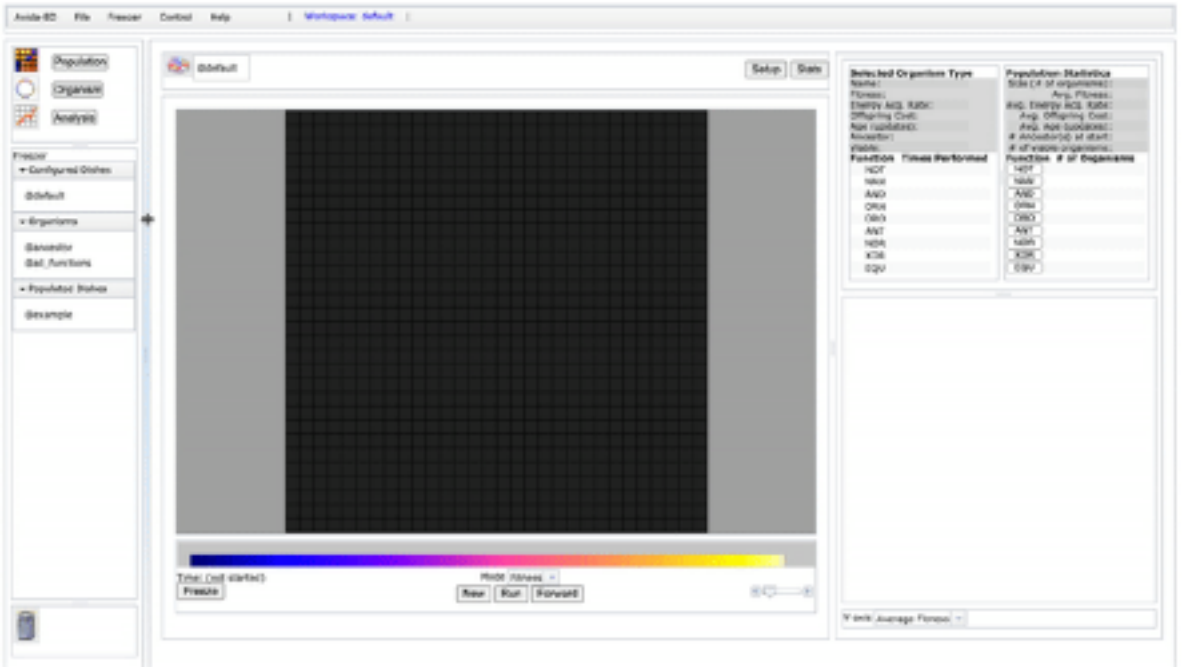
Many thanks to Diane Blackwood, my advisor, without whom I would not have been able to achieve as much as I did. And thank you to all fellow WAVES mentors and participants! This work is supported through Active LENS: Learning Evolution and the Nature of Science using Evolution in Action (NSF IUSE #1432563). Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

Acknowledgements


Avida-ED
File
Freezer
Control
Help
Workspace: "Default" save status: saved
Avida-ED 4.0.02 Beta
Start Setup
Population
Organism
Analysis
Freezer
Configured Dishes
@default
Organisms
@all_functions
@ancestor
Populated Dishes
@example
@default
Selected Organism Type
Name:
Fitness:
Energy Acq. Rate:
Offspring Cost:
Age (updates):
Ancestor:
Viable:
Function
Times Performed
NOT
NAN
AND
ORN
ORO
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
ORO
ANT
NOR
XOR
EQU
Time: (not started)
Node:
Fitness
Freeze
New
Run
Forward
Y-axis
Avg. Fitness
Options
Combined
Pause Run at
Update
1000
when checked

<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, but did not work on touch devices.


Population
Organism
Analysis
Setup
Stats
Freeze
Run
Forward

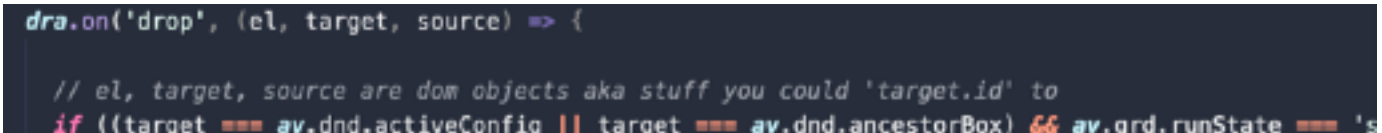
<Figure 2. Avida-ED 3>

Dragula Drag & Drop

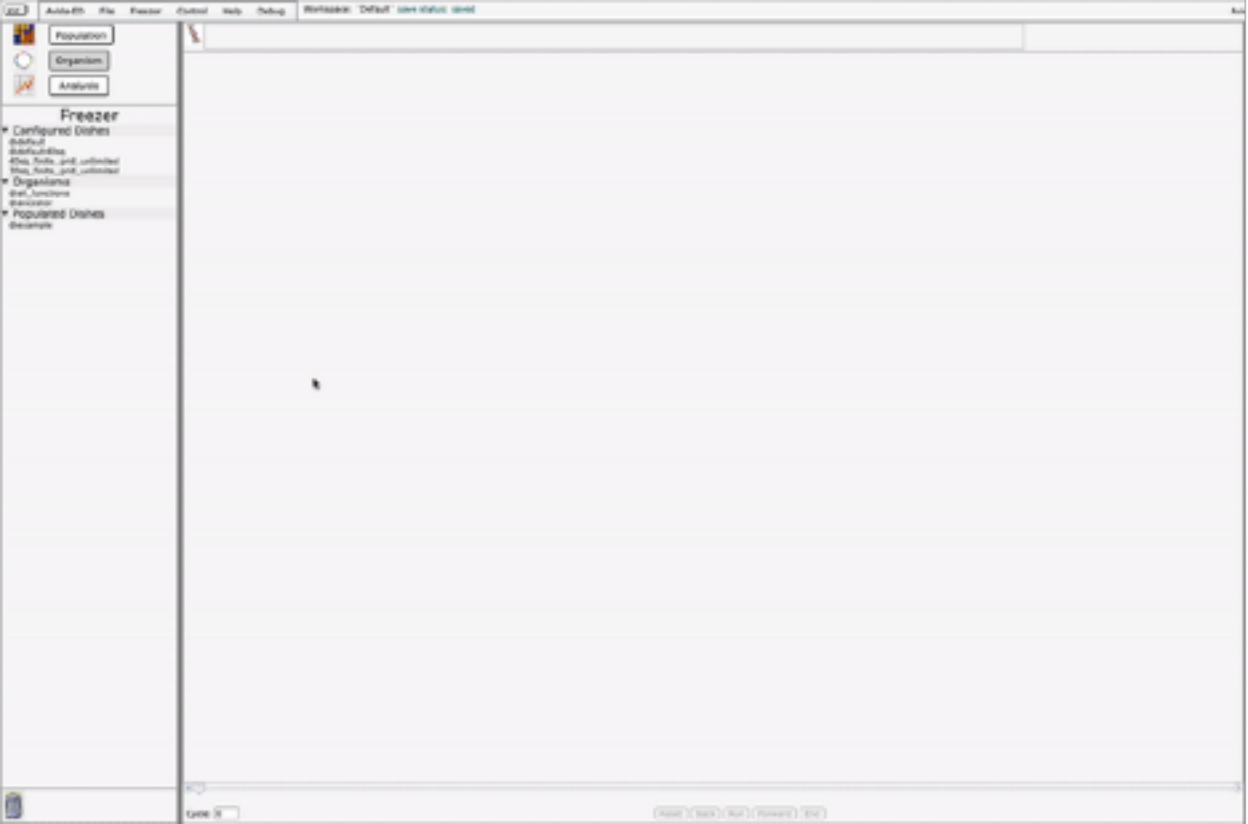
```
var population_colInfo = widthOfNav + "px 3px " + "auto 3px " + rightSideWidth;
var organism_colInfo = widthOfNav + "px 3px " + "auto 3px " + rightSideWidth;
var analysis_colInfo = widthOfNav + "px 3px auto";
$('.all2lft').css("grid-template-columns", analysis_colInfo); /* yemi: you need to resize 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 Avida-ED 4 on Touch Device>


Population
Organism
Analysis
Freezer
Configured Dishes
Organisms
Populated Dishes

<< Avida-ED File Freezer Control Help Debug Workspace: `Default` save status: sav

Population

Organism

Analysis

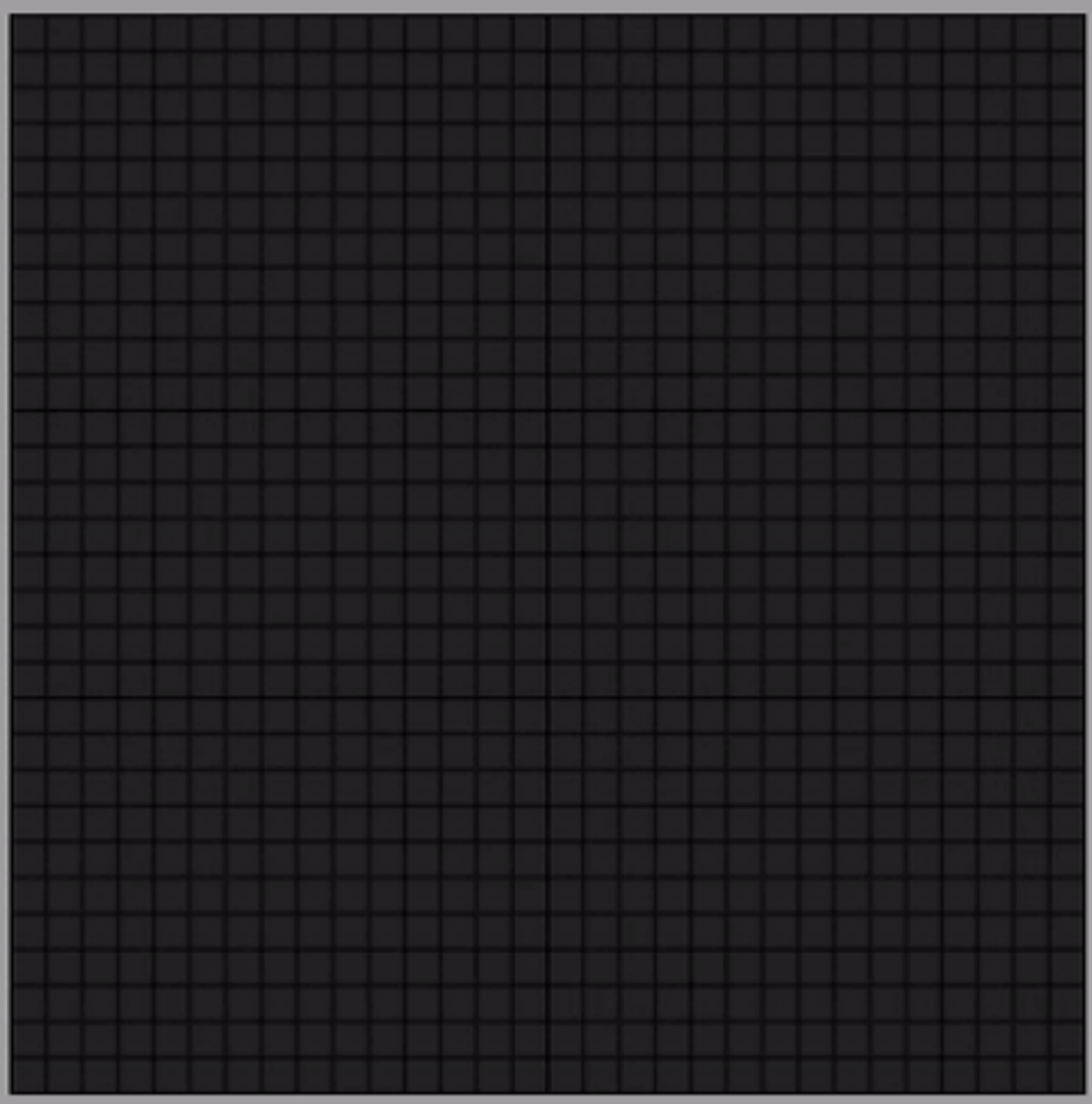
Freezer

▼ Configured Dishes
@default
@default40sq
40sq_finite_grid_unlimited
30sq_finite_grid_unlimited

▼ Organisms
@all_functions
@ancestor

▼ Populated Dishes
@example

@default

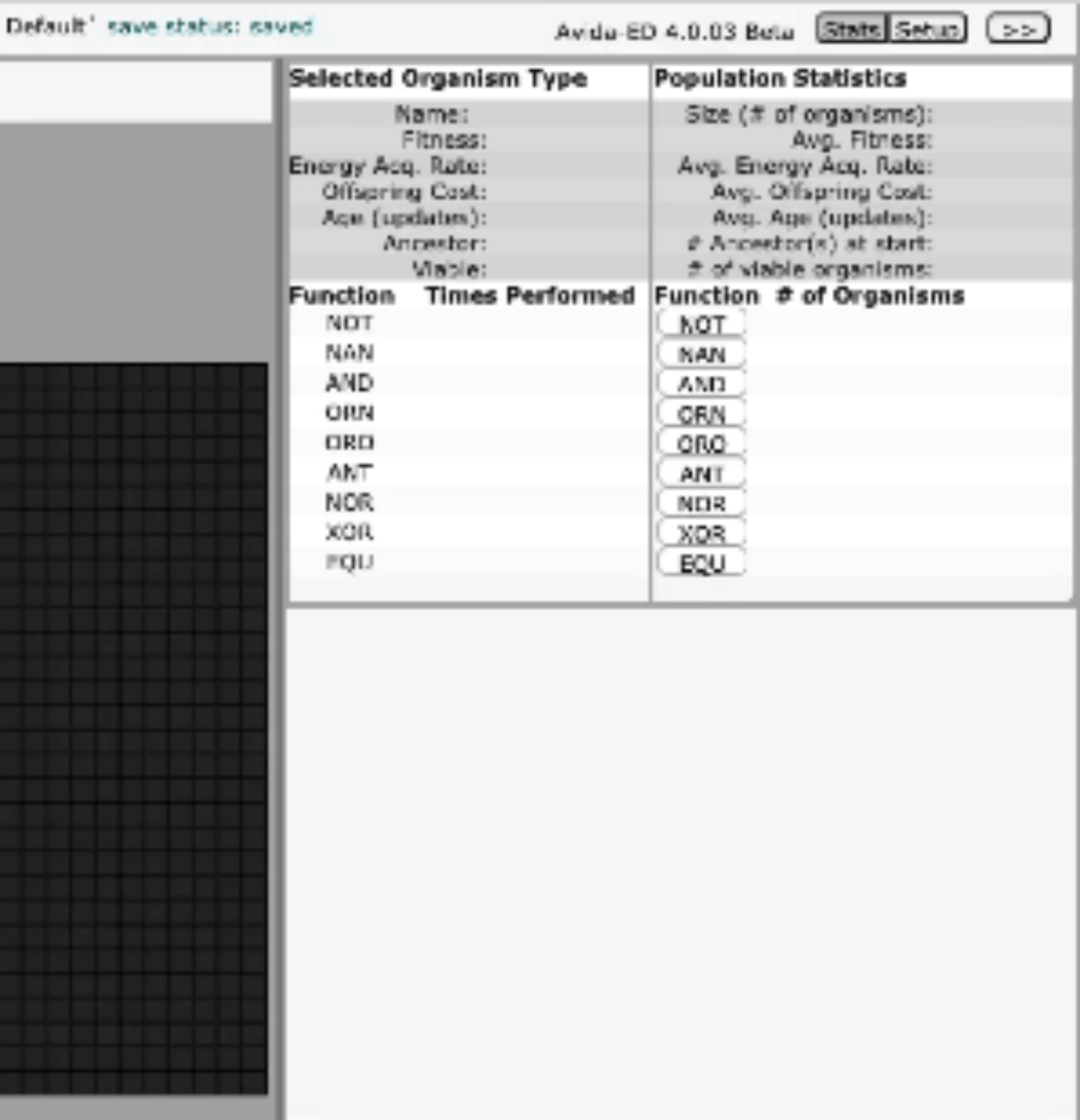


Time: (not started)

Freeze

Mode: Fitness

New Run Forward


Default' save status: saved
Avida-ED 4.0.03 Beta
Stats
Setup
>>
Selected Organism Type
Name:
Fitness:
Energy Acq. Rate:
Offspring Cost:
Age (updates):
Ancestor:
Viable:
Function Times Performed
NOT
NAN
AND
ORN
ORO
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
ORO
ANT
NOR
XOR
EQU

1. Capture the horizontal mouse position

2. Calculate 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 css of the grid

1. Set up the Dragula Drag&Drop Engine

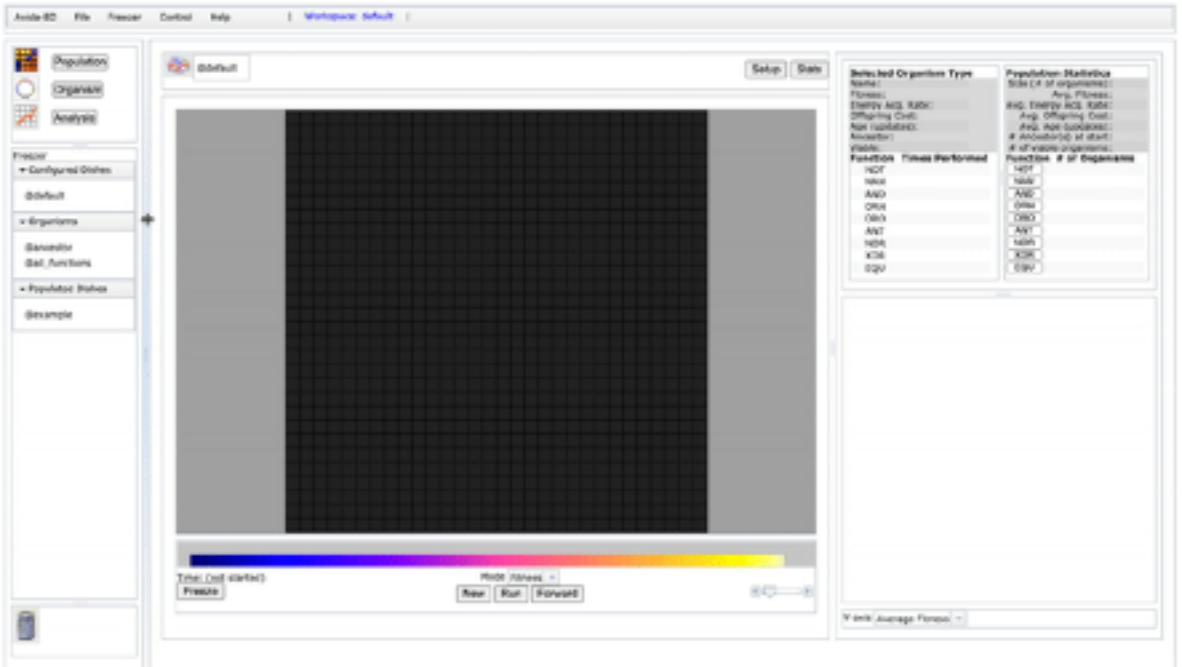
2. Capture the ‘drop’ action

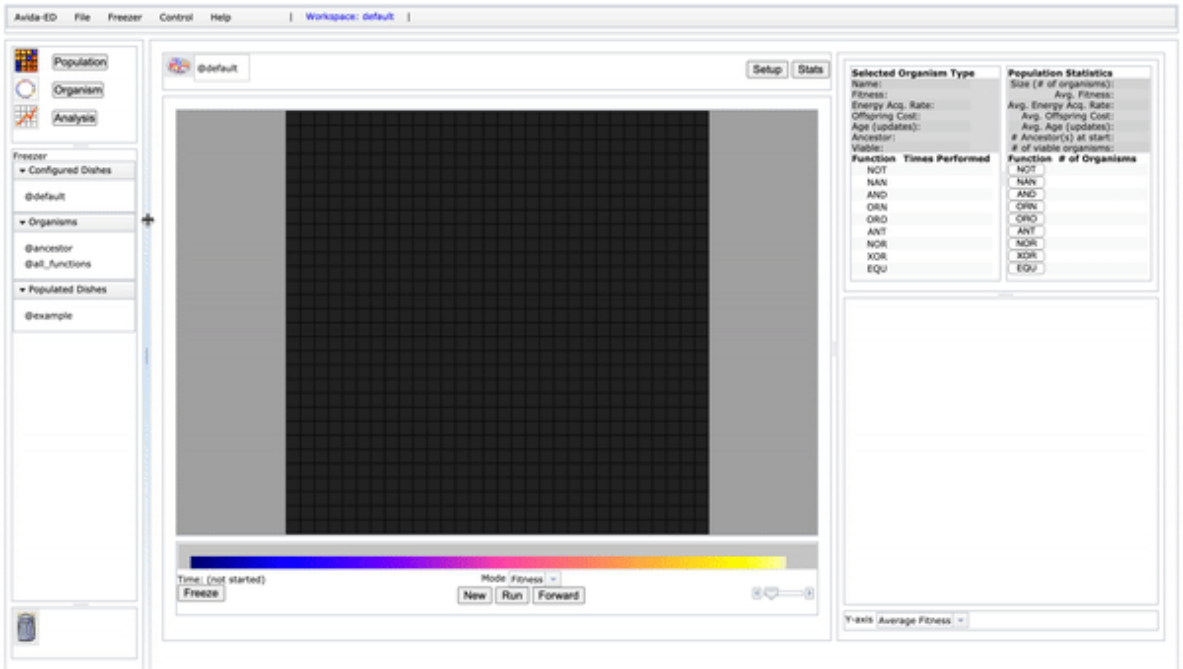
```
// when a configured dish is added to the config box
av.dnd.landActiveConfig = function (el, target, source) {
  'use strict';
  av.dnd.configFlag = 'normal';

  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];
```

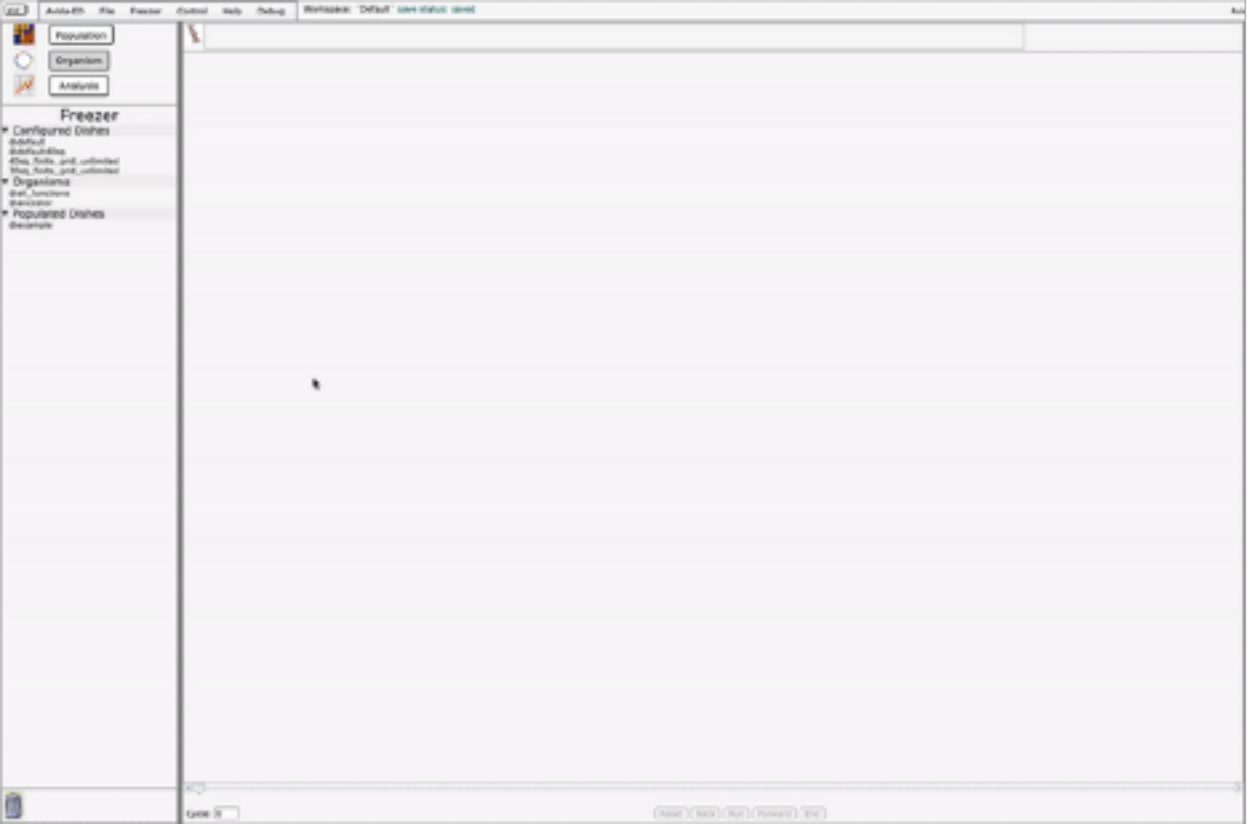
3. Handle the ‘drop’ action

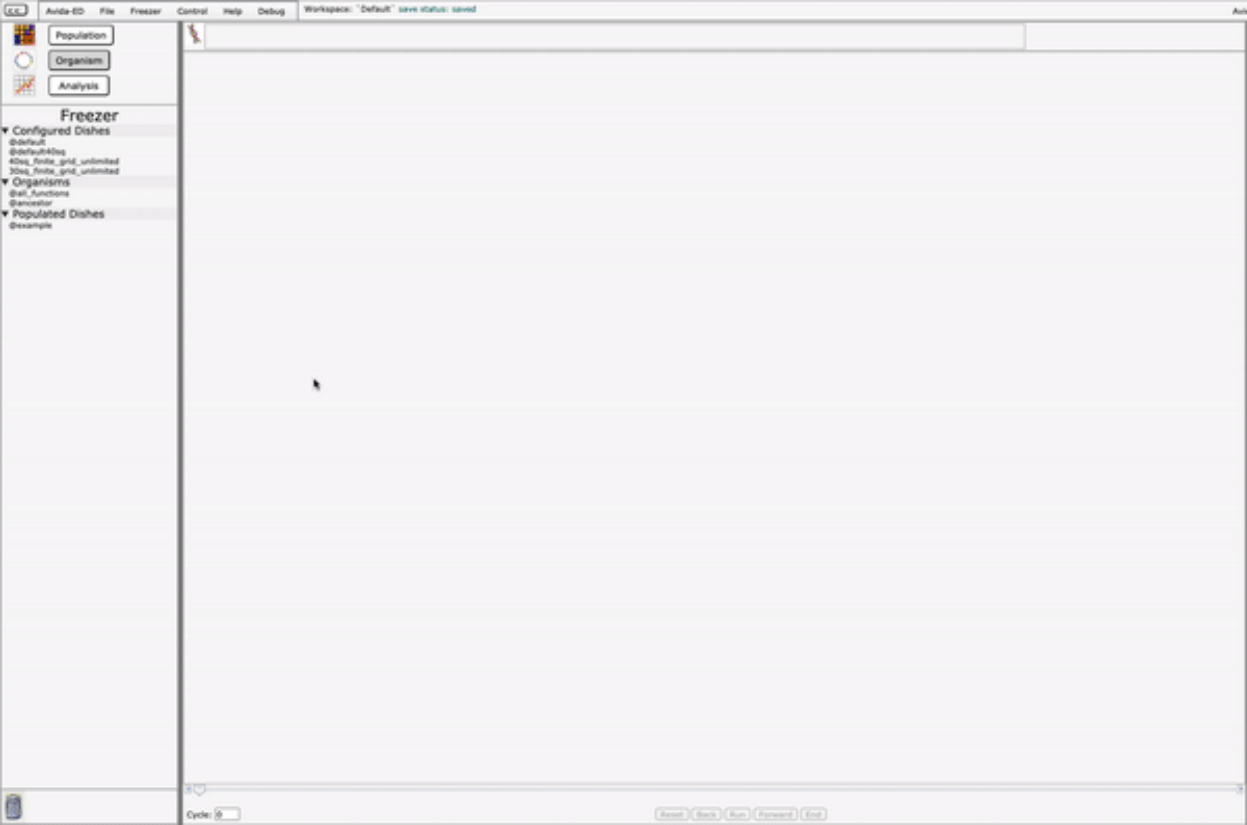
3.a. Update the DOM

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


Population
Organism
Analysis
Setup
Stats
Freeze
Run
Forward


Avida-ED
File
Freezer
Control
Help
Workspace: default
Population
Organism
Analysis
Freezer
Configured Dishes
@default
Organisms
@ancestor
@all_functions
Populated Dishes
@example
@default
Setup
Stats
Selected Organism Type
Name:
Fitness:
Energy Acq. Rate:
Offspring Cost:
Age (updates):
Ancestor:
Viable:
Function Times Performed
NOT
NAN
AND
ORN
ORO
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
ORO
ANT
NOR
XOR
EQU
Time: (not started)
Mode: Fitness
Freeze
New
Run
Forward
Y-axis: Average Fitness


Population
Organism
Analysis
Freezer
Configured Dishes
Organisms
Populated Dishes


Avida-ED
File
Freezer
Control
Help
Debug
Workspace: "Default" save status: saved
Population
Organism
Analysis
Freezer
Configured Dishes
@default
@default40sq
40sq_finite_grid_unlimited
30sq_finite_grid_unlimited
Organisms
@all_functions
@ancestor
Populated Dishes
@example
Cycle: 0
Reset
Back
Run
Forward
End

<< Avida-ED File Freezer Control Help Debug

Workspace: `Default` save status: sav

Population

Organism

Analysis

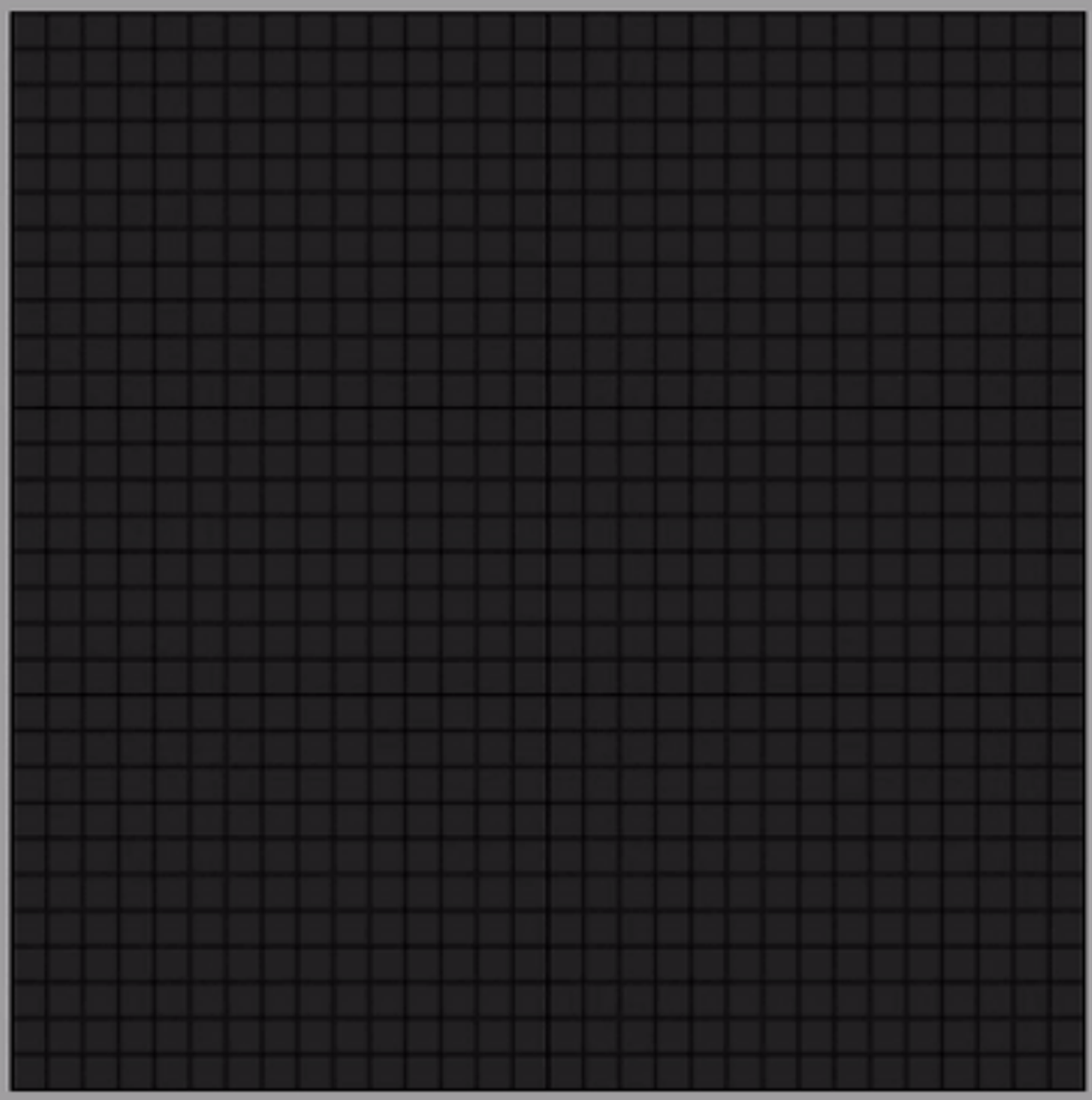
Freezer

▼ Configured Dishes
- @default
- @default40sq
- 40sq_finite_grid_unlimited
- 30sq_finite_grid_unlimited

▼ Organisms
- @all_functions
- @ancestor

▼ Populated Dishes
- @example

@default

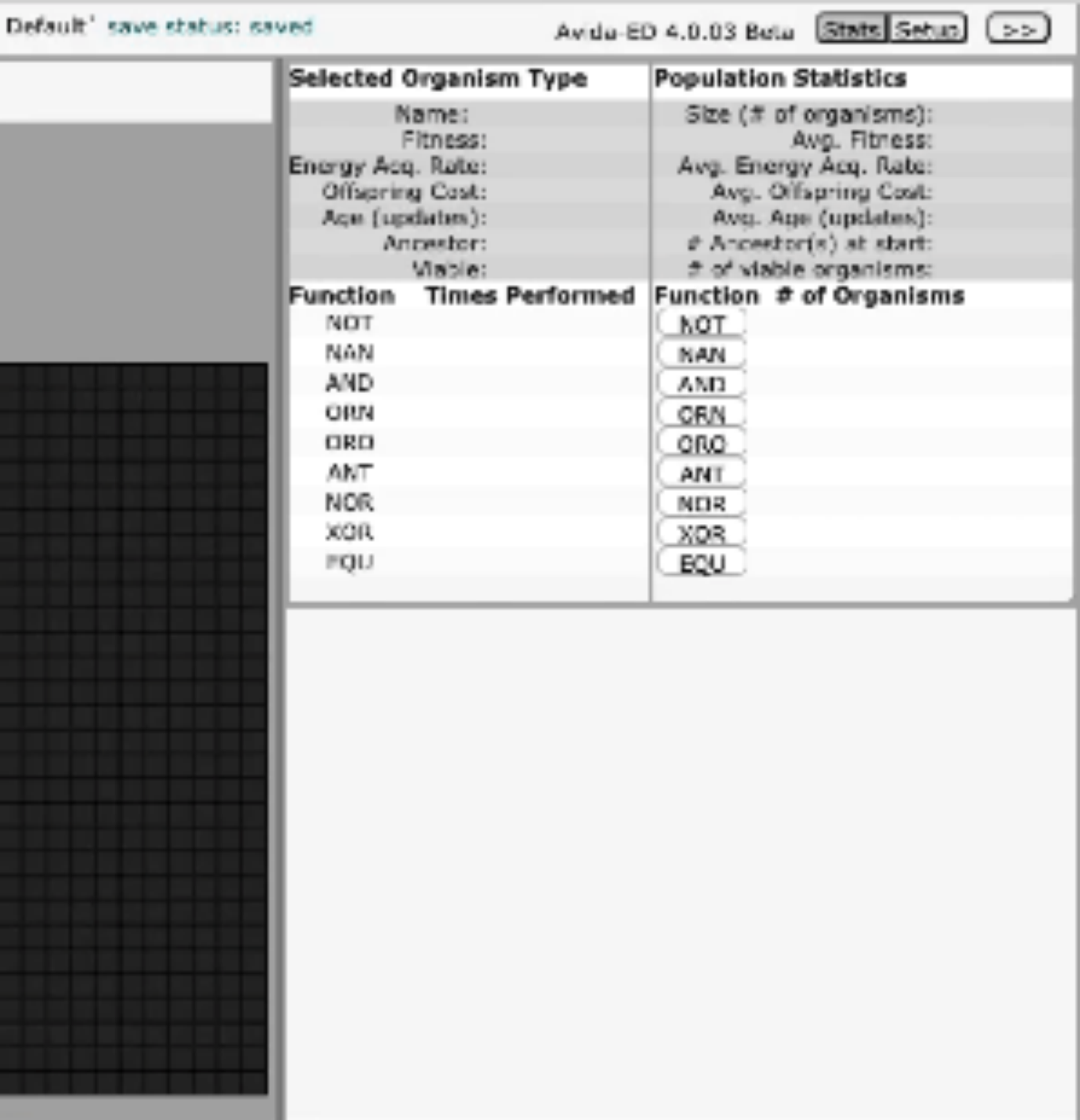


Time: (not started)

Freeze

Mode: Fitness

New Run Forward

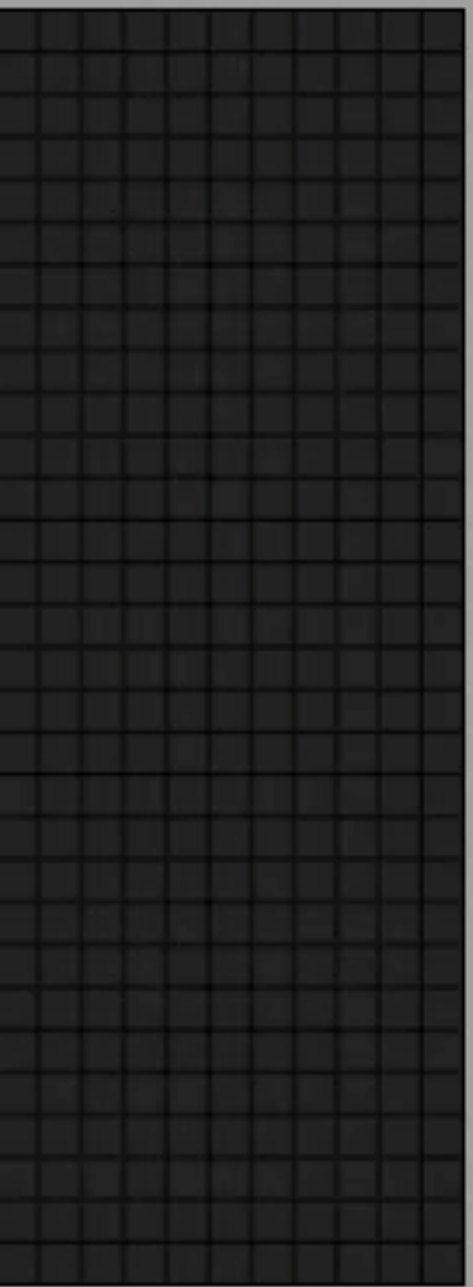

Default' save status: saved
Avida-ED 4.0.03 Beta
Stats
Setup
>>
Selected Organism Type
Name:
Fitness:
Energy Acq. Rate:
Offspring Cost:
Age (updates):
Ancestor:
Viable:
Function Times Performed
NOT
NAN
AND
ORN
ORO
ANT
NOR
XOR
EQU
Population Statistics
Size (# of organisms):
Avg. Fitness:
Avg. Energy Acq. Rate:
Avg. Offspring Cost:
Avg. Age (updates):
Ancestor(s) at start:
of viable organisms:
Function # of Organisms
NOT
NAN
AND
ORN
ORO
ANT
NOR
XOR
EQU

Default` save status: saved

Avida-ED 4.0.03 Beta

Stats | Setup

>>



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

Function	Times Performed
NOT	
NAN	
AND	
ORN	
ORO	
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	
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

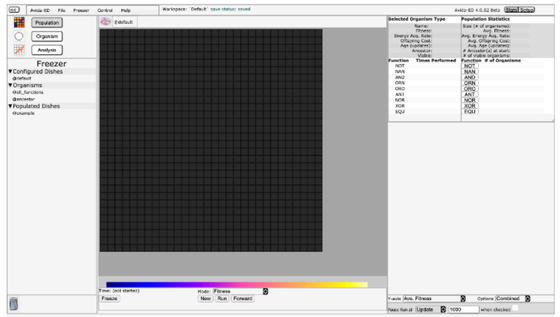
Avida-ED…

- 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

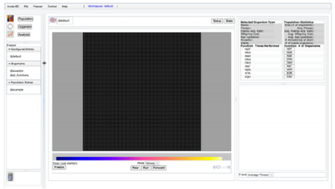
1.Implement dragbars to allow for freely resizable windows.

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



<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, but did not work on touch devices.



<Figure 2. Avida-ED 3>

Methodology

Dragbars

reSizePageParts.js

```
$(document).on('mousemove touchmove', function(e){
  av.grd.drawGridSetupFn(); // yemi: redraw the grid
  av.ani.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 horizontal 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 480px (right
var percentage = (x / widthAvailable);
var widthOfNav = widthAvailable * percentage;
```

2. Calculate the layout grid column sizes

```
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";
$('.all2ft').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.organIcon || target ==
      return true;
    }
    if (source === av.dnd.activeConfig && (target === av.dnd.frConfig || target ===
      return true;
    }
```

1. Set up the Dragula Drag&Drop Engine

```
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.frr.actConfig.actDomid = domid;
  av.frr.actConfig.name = el.textContent;
  av.frr.actConfig.frDomid = source.id;
  av.frr.actConfig.dir = av.frr.dir(av.frr.actConfig.actDomid);
```

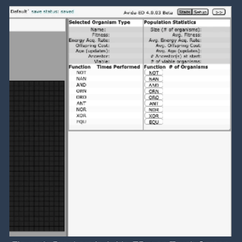
3. Handle the 'drop' action

3.a. Update the DOM

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

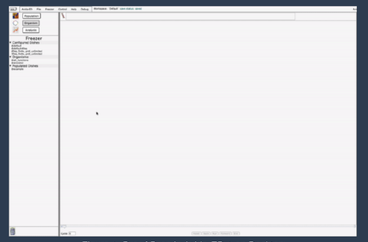
Results

Dragbars

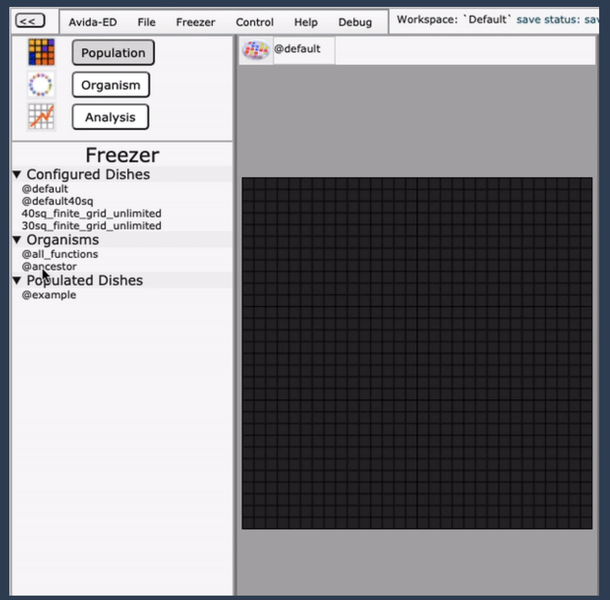


<Figure 3. Dragbars in Avida-ED 4 on Touch Screen>

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

- I was constantly astounded by, and am still being astounded by the variety of ways that JavaScript, and JQuery, and the dollar signs (who knew that $ had so many meanings) work together but at times are frustratingly incompatible with each other. They were the secret sauce to the magic on the screen, in the end, though.

- Something as simple as a dragbar or drag&drop, is rather complicated in its inner workings!

Steps I took to implement the Project:

1. 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 that make up a drawbar, and mocked up a dragbar apart from the main codebase.

4. I copied, pasted, modified my poc dragbar into the codebase, and tweaked several things to make it actually compatible with the rest of the Avida-ED code and layout (Hello, maquetta grid!)

The Importance of Proof of Concept

- It is a great chance for you get yourself familiar with the Javascript functions that will be handy for your task.

- 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 software.

- It gives you a template, a kind of jumping off point to get you going (and with confidence!) when you finally touch the codebase. Fear not!

Approach Code Like a Surgeon

If you approach code like a surgeon, and are very aware of every change you make precisely, you will save much time and energy later down the road when you might need to back track to locate the origin of some error blind sighted! (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 thank you to all fellow WAVES mentors and participants! This work is supported through Active LENS: Learning Evolution and the Nature of Science using Evolution in Action (NSF IUSE #1432563). Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.