Slides, videos, links and more:
https://github.com/physicell-training/ws2021

Session 1: Working with PhysiCell Projects

Paul Macklin, Ph.D.
@MathCancer

PhysiCell Project

July 15, 2021

LUDDY
SCHOOL OF INFORMATICS, COMPUTING, AND ENGINEERING

PhysiCell Project
PhysiCell.org
@PhysiCell

Goals

- Learn how to work with sample projects
  - Get a list of sample projects
  - Populate a project
  - Look at typical project structure
  - Modify settings
  - Compile and run a populated project
  - See typical model outputs
  - Clear out data and reset

Sample projects

- It's inefficient (and a little insane) to code new projects *entirely* from scratch.
- So, we provide sample projects:
  - 2D/3D template project
  - Cancer models
  - Synthetic multicellular systems
  - Viral dynamics in tissue
  - and more …

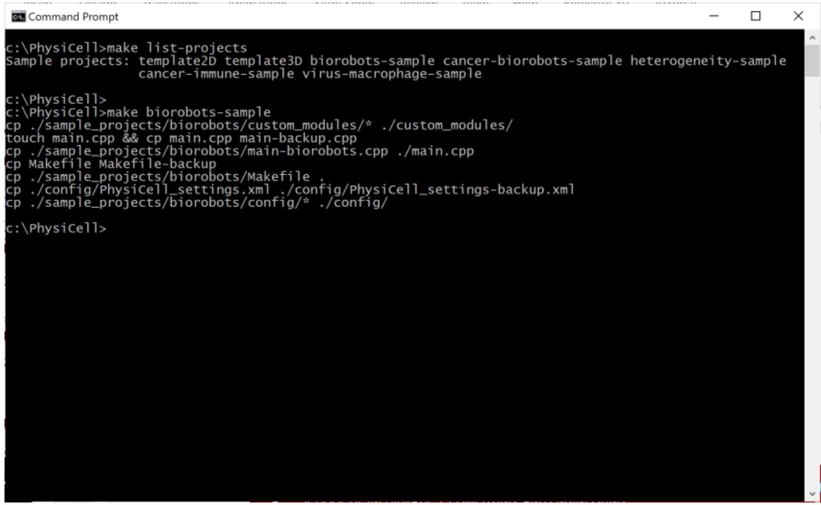
- **make [project-name]:** populate a sample project (puts all the source files where they belong)
  - Then use **make** to compile it
- **make data-cleanup**: clean up the output data
- **make reset**: return to a "clean slate" (depopulate the project)
- **make list-projects:** display all available sample projects

Documentation: User Guide Sections 6, 7.

PhysiCell Project Essentials (1)

- Each PhysiCell release includes sample projects. To list them:
  - `make list-projects`

- Your first step is to **populate a project.**
  - `make <project_name>`
  - Let's use biorobots-sample:
    - `make biorobots-sample`
  - This copies source code, a tailored make file, and configuration files


Command Prompt

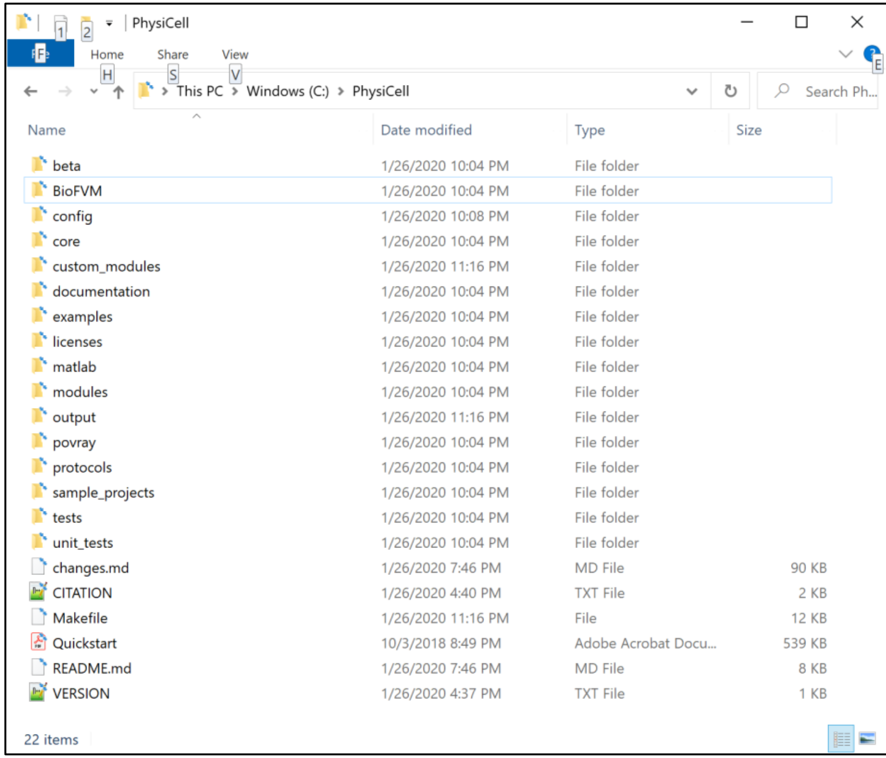
```
c:\PhysiCell>make list-projects
Sample projects: template2D template3D biorobots-sample cancer-biorobots-sample heterogeneity-sample
                 cancer-immune-sample virus-macrophage-sample

c:\PhysiCell>
c:\PhysiCell>make biorobots-sample
cp ./sample_projects/biorobots/custom_modules/* ./custom_modules/
touch main.cpp && cp main.cpp main-backup.cpp
cp ./sample_projects/biorobots/main-biorobots.cpp ./main.cpp
cp Makefile Makefile-backup
cp ./sample_projects/biorobots/Makefile .
cp ./config/PhysiCell_settings.xml ./config/PhysiCell_settings-backup.xml
cp ./sample_projects/biorobots/config/* ./config/

c:\PhysiCell>
```

Let's look at the project structure …

Project directory structure

- (key) directories:
  - **./ (root):** main source, Makefile, and executable go here
  - **./beta**: for beta-testing (don't use)
  - **./BioFVM:** diffusion solver
  - **./config:** configuration files
  - **./core:** PhysiCell core functions
  - **./custom_modules:** put custom code for your project here.
  - **./documentation:** user guide, etc.
  - **./examples:** deprecated
  - **./licenses:** yep
  - **./matlab:** scripts and functions to load data in matlab
  - **./modules:** standard add-ons for PhysiCell
  - **./output:** where data are stored (by default, but can be changed)
  - **./povray:** deprecated
  - **./protocols:** instructions mostly for maintainers (e.g., release protocols)
  - **./sample_projects:** where we add sample projects
  - **./tests:** for automated testing (WIP)
  - **./unit_tests:** for automated testing (WIP)

Most of your work will be in the red directories

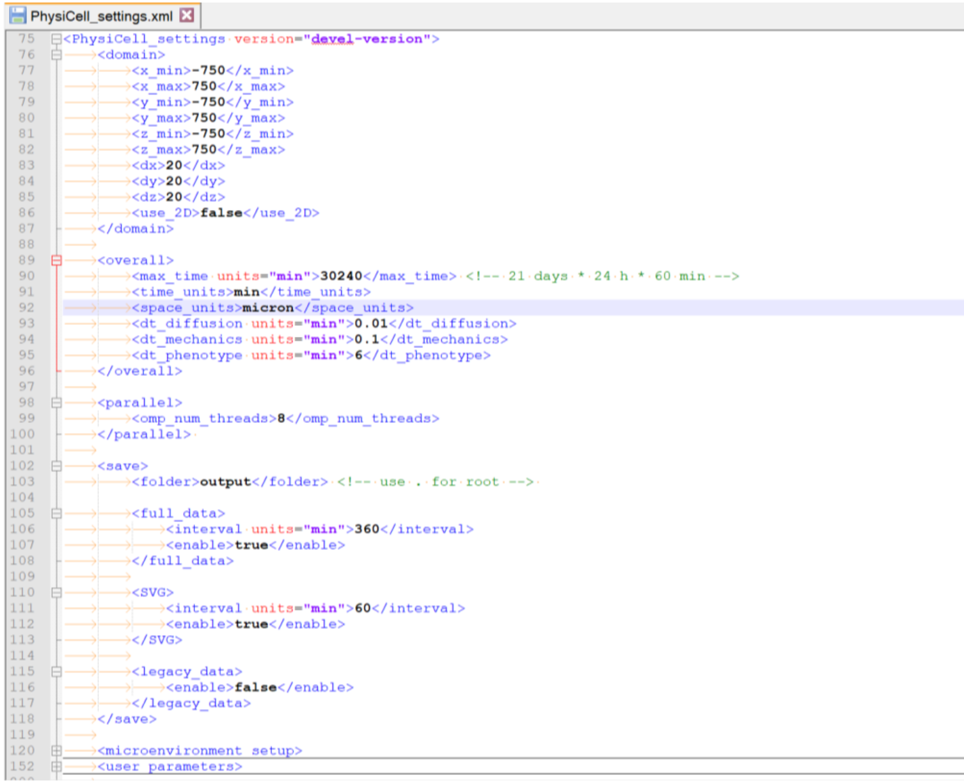


Name	Date modified	Type	Size
beta	1/26/2020 10:04 PM	File folder	
BioFVM	1/26/2020 10:04 PM	File folder	
config	1/26/2020 10:08 PM	File folder	
core	1/26/2020 10:04 PM	File folder	
custom_modules	1/26/2020 11:16 PM	File folder	
documentation	1/26/2020 10:04 PM	File folder	
examples	1/26/2020 10:04 PM	File folder	
licenses	1/26/2020 10:04 PM	File folder	
matlab	1/26/2020 10:04 PM	File folder	
modules	1/26/2020 10:04 PM	File folder	
output	1/26/2020 11:16 PM	File folder	
povray	1/26/2020 10:04 PM	File folder	
protocols	1/26/2020 10:04 PM	File folder	
sample_projects	1/26/2020 10:04 PM	File folder	
tests	1/26/2020 10:04 PM	File folder	
unit_tests	1/26/2020 10:04 PM	File folder	
changes.md	1/26/2020 7:46 PM	MD File	90 KB
CITATION	1/26/2020 4:40 PM	TXT File	2 KB
Makefile	1/26/2020 11:16 PM	File	12 KB
Quickstart	10/3/2018 8:49 PM	Adobe Acrobat Docu...	539 KB
README.md	1/26/2020 7:46 PM	MD File	8 KB
VERSION	1/26/2020 4:37 PM	TXT File	1 KB

22 items

Project structure: config files

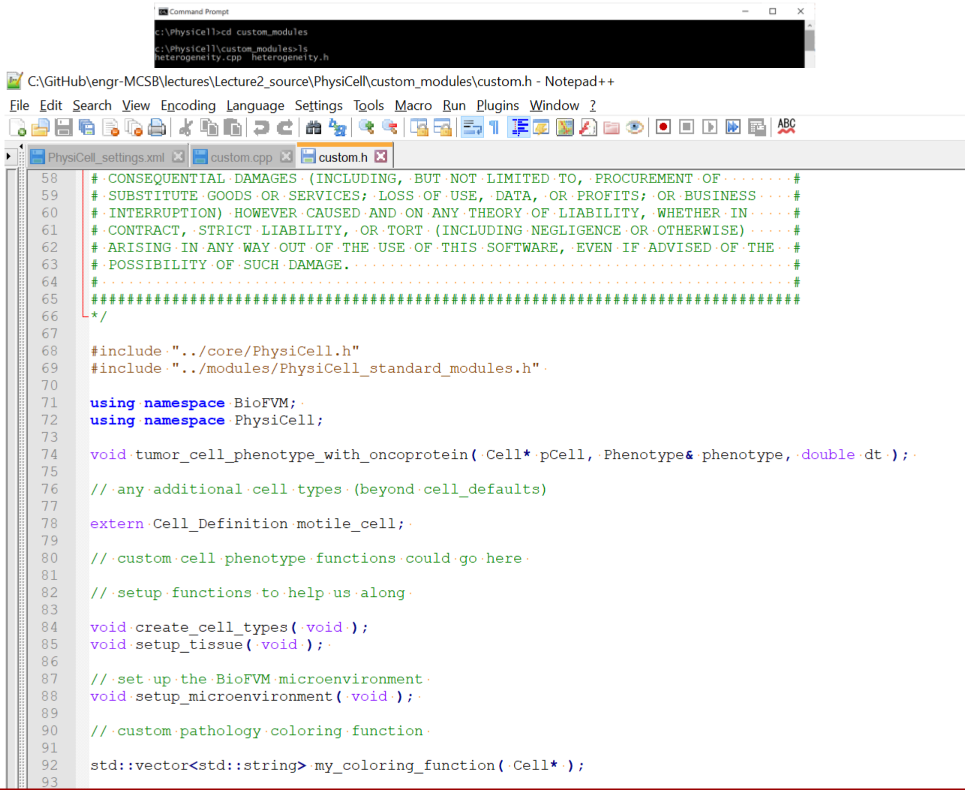
- Configuration files (XML)
  - **domain:** domain size and resolution
  - **overall:** general options
    - ♦ Final simulation time
    - ♦ Time step sizes
  - **parallel:** parallelization options
    - ♦ Number of threads
  - **save:** save options
    - ♦ Save where?
    - ♦ Save SVGs? (how often?)
    - ♦ Save full data? (how often?)
    - ♦ Save legacy data (don't)
  - **microenvironment_setup:** diffusion settings
    - ♦ more later
  - **cell_definitions:** define different cell types and starting parameters
    - ♦ more later
  - **user_parameters:** simulation-specific settings
    - ♦ more later


PhysiCell_settings.xml

```xml
<PhysiCell_settings version="devel-version">
	<domain>
		<x_min>-750</x_min>
		<x_max>750</x_max>
		<y_min>-750</y_min>
		<y_max>750</y_max>
		<z_min>-750</z_min>
		<z_max>750</z_max>
		<dx>20</dx>
		<dy>20</dy>
		<dz>20</dz>
		<use_2D>false</use_2D>
	</domain>
	
	<overall>
		<max_time units="min">30240</max_time> <!-- 21 days * 24 h * 60 min -->
		<time_units>min</time_units>
		<space_units>micron</space_units>
		<dt_diffusion units="min">0.01</dt_diffusion>
		<dt_mechanics units="min">0.1</dt_mechanics>
		<dt_phenotype units="min">6</dt_phenotype>
	</overall>
	
	<parallel>
		<omp_num_threads>8</omp_num_threads>
	</parallel>
	
	<save>
		<folder>output</folder> <!-- use . for root -->

		<full_data>
			<interval units="min">360</interval>
			<enable>true</enable>
		</full_data>
		
		<SVG>
			<interval units="min">60</interval>
			<enable>true</enable>
		</SVG>
		
		<legacy_data>
			<enable>false</enable>
		</legacy_data>
	</save>
	
	<microenvironment_setup>
	<user_parameters>
```

Project structure: custom modules

- Custom Modules
  - Setup functions
  - Cell definitions
  - Custom functions
  - any other modeling
  - Custom coloring functions



Command Prompt

```
c:\PhysiCell>cd custom_modules

c:\PhysiCell\custom_modules>ls
heterogeneity.cpp  heterogeneity.h
```

C:\GitHub\engr-MCSB\lectures\Lecture2_source\PhysiCell\custom_modules\custom.h - Notepad++

File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
PhysiCell_settings.xml
custom.cpp
custom.h

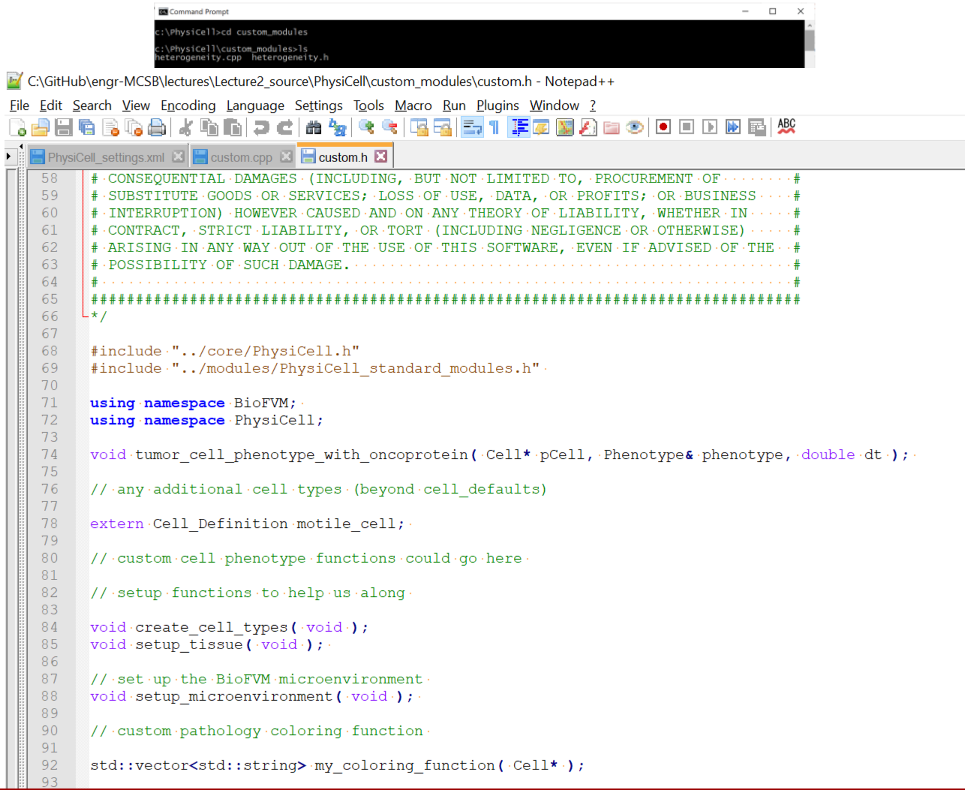
```cpp
# CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF       #
# SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS   #
# INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN    #
# CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE)    #
# ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE  #
# POSSIBILITY OF SUCH DAMAGE.                                                 #
#                                                                             #
###############################################################################
*/

#include "../core/PhysiCell.h"
#include "../modules/PhysiCell_standard_modules.h"

using namespace BioFVM;
using namespace PhysiCell;

void tumor_cell_phenotype_with_oncoprotein( Cell* pCell, Phenotype& phenotype, double dt );

// any additional cell types (beyond cell_defaults)

extern Cell_Definition motile_cell;

// custom cell phenotype functions could go here

// setup functions to help us along

void create_cell_types( void );
void setup_tissue( void );

// set up the BioFVM microenvironment
void setup_microenvironment( void );

// custom pathology coloring function

std::vector<std::string> my_coloring_function( Cell* );
```

LUDDY
SCHOOL OF INFORMATICS, COMPUTING, AND ENGINEERING

PhysiCell Project
PhysiCell.org
@PhysiCell

Project structure: custom modules

- Custom Modules
  - Any user-defined globals (at top)
    - Declared cell types
  - Setup functions
    - `create_cell_types()`
      - Do all setup on all cell types
        - Adjust phenotype
        - Add / adjust custom data
        - Set functions
    - `setup_tissue()`
      - Place initial cells in microenvironment
      - Modify each cell as needed
  - Custom functions
  - any other modeling
  - Custom coloring functions



```
c:\PhysiCell>cd custom_modules

c:\PhysiCell\custom_modules>ls
heterogeneity.cpp  heterogeneity.h
```

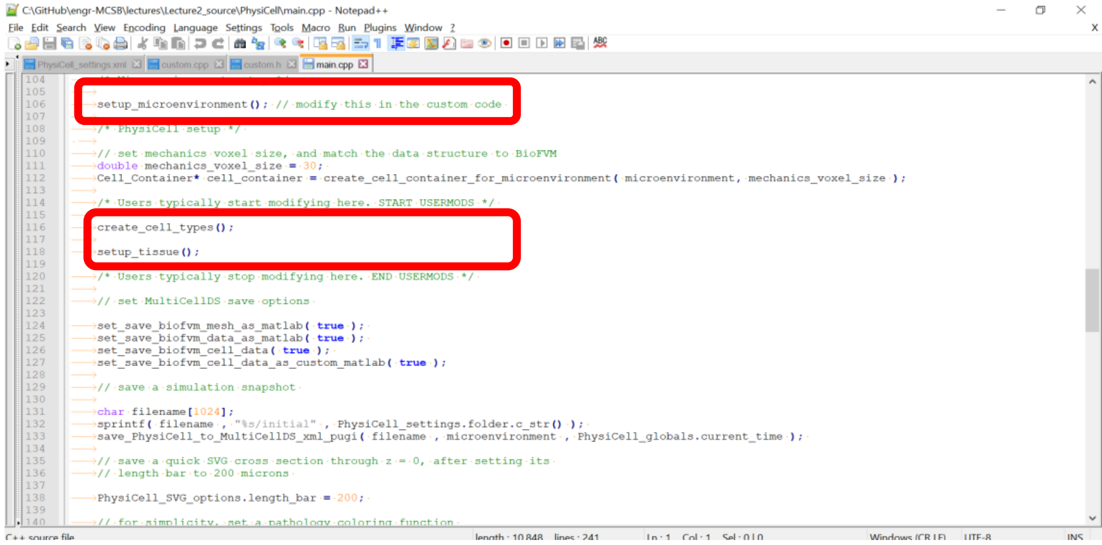
C:\GitHub\engr-MCSB\lectures\Lecture2_source\PhysiCell\custom_modules\custom.h - Notepad++

File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
PhysiCell_settings.xml
custom.cpp
custom.h

```
# CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF       #
# SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS   #
# INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN    #
# CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE)    #
# ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE #
# POSSIBILITY OF SUCH DAMAGE.                                                #
#                                                                            #
##############################################################################
*/

#include "../core/PhysiCell.h"
#include "../modules/PhysiCell_standard_modules.h"

using namespace BioFVM;
using namespace PhysiCell;

void tumor_cell_phenotype_with_oncoprotein( Cell* pCell, Phenotype& phenotype, double dt );

// any additional cell types (beyond cell_defaults)

extern Cell_Definition motile_cell;

// custom cell phenotype functions could go here

// setup functions to help us along

void create_cell_types( void );
void setup_tissue( void );

// set up the BioFVM microenvironment
void setup_microenvironment( void );

// custom pathology coloring function

std::vector<std::string> my_coloring_function( Cell* );

```

Project structure: main.cpp

- **main.cpp**
  - (in the root directory)
  - calls the setup functions

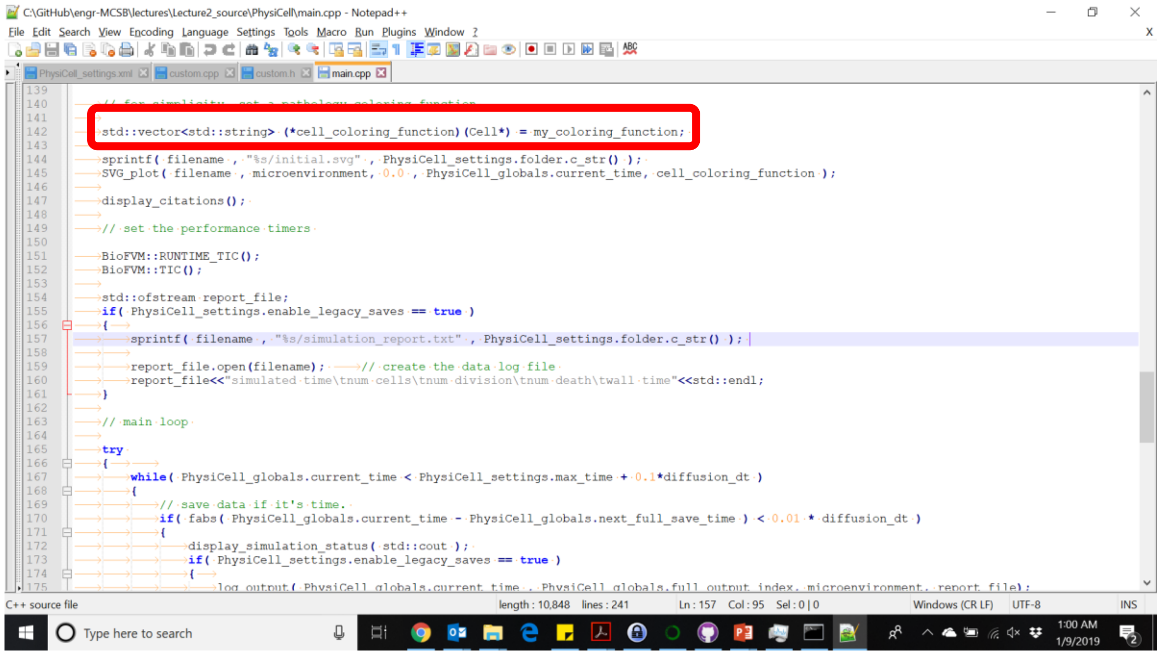

C:\GitHub\engr-MCSB\lectures\Lecture2_source\PhysiCell\main.cpp - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
PhysiCell_settings.xml
custom.cpp
custom.h
main.cpp

```cpp

	setup_microenvironment(); // modify this in the custom code 

	/* PhysiCell setup */ 
	
	// set mechanics voxel size, and match the data structure to BioFVM
	double mechanics_voxel_size = 30; 
	Cell_Container* cell_container = create_cell_container_for_microenvironment( microenvironment, mechanics_voxel_size );
	
	/* Users typically start modifying here. START USERMODS */ 
	
	create_cell_types();
	
	setup_tissue();

	/* Users typically stop modifying here. END USERMODS */ 
	
	// set MultiCellDS save options 

	set_save_biofvm_mesh_as_matlab( true ); 
	set_save_biofvm_data_as_matlab( true ); 
	set_save_biofvm_cell_data( true ); 
	set_save_biofvm_cell_data_as_custom_matlab( true );
	
	// save a simulation snapshot 
	
	char filename[1024];
	sprintf( filename , "%s/initial" , PhysiCell_settings.folder.c_str() ); 
	save_PhysiCell_to_MultiCellDS_xml_pugi( filename , microenvironment , PhysiCell_globals.current_time ); 
	
	// save a quick SVG cross section through z = 0, after setting its 
	// length bar to 200 microns 

	PhysiCell_SVG_options.length_bar = 200; 

	// for simplicity, set a pathology coloring function 
```

C++ source file
length : 10,848 lines : 241
Ln : 1 Col : 1 Sel : 0 | 0
Windows (CR LF) UTF-8 INS

Project structure: main.cpp (continued)

- **`main.cpp`**
  - set coloring function



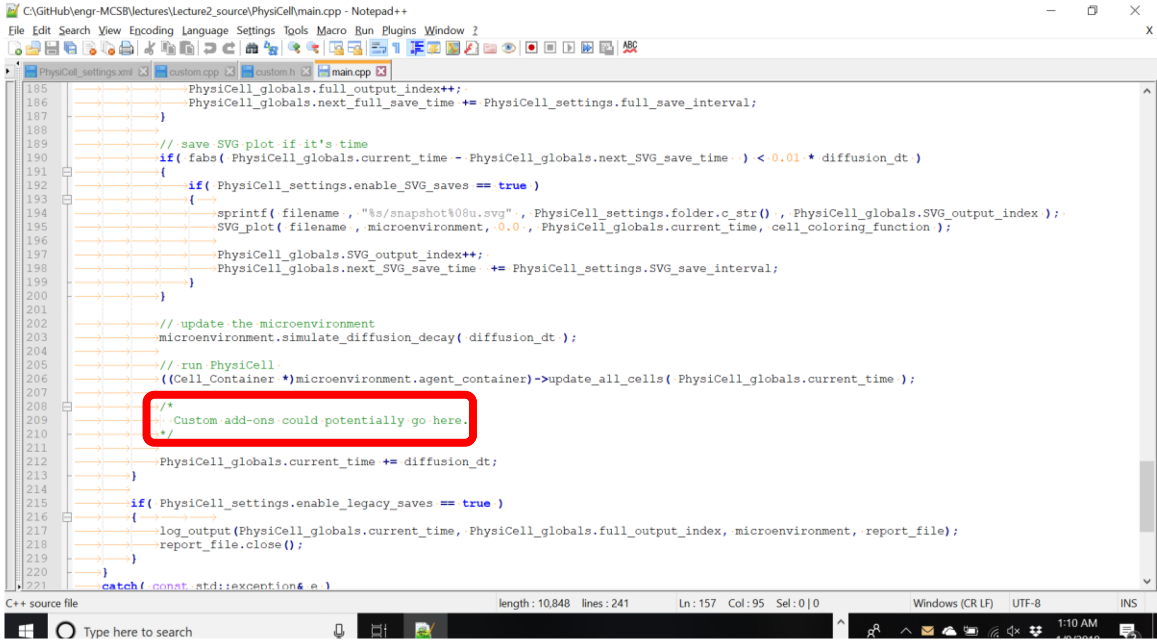
C:\GitHub\engr-MCSB\lectures\Lecture2_source\PhysiCell\main.cpp - Notepad++

```cpp
	// for simplicity, set a pathology coloring function

	std::vector<std::string> (*cell_coloring_function)(Cell*) = my_coloring_function;

	sprintf( filename , "%s/initial.svg" , PhysiCell_settings.folder.c_str() );
	SVG_plot( filename , microenvironment, 0.0 , PhysiCell_globals.current_time, cell_coloring_function );

	display_citations();

	// set the performance timers

	BioFVM::RUNTIME_TIC();
	BioFVM::TIC();

	std::ofstream report_file;
	if( PhysiCell_settings.enable_legacy_saves == true )
	{
		sprintf( filename , "%s/simulation_report.txt" , PhysiCell_settings.folder.c_str() );

		report_file.open(filename); 	// create the data log file
		report_file<<"simulated time\tnum cells\tnum division\tnum death\twall time"<<std::endl;
	}

	// main loop

	try
	{
		while( PhysiCell_globals.current_time < PhysiCell_settings.max_time + 0.1*diffusion_dt )
		{
			// save data if it's time.
			if( fabs( PhysiCell_globals.current_time - PhysiCell_globals.next_full_save_time ) < 0.01 * diffusion_dt )
			{
				display_simulation_status( std::cout );
				if( PhysiCell_settings.enable_legacy_saves == true )
				{
					log_output( PhysiCell_globals.current_time , PhysiCell_globals.full_output_index, microenvironment, report_file);
```

C++ source file   length : 10,848   lines : 241   Ln : 157   Col : 95   Sel : 0 | 0   Windows (CR LF)   UTF-8   INS

Project structure: main.cpp (continued)

- **`main.cpp`**
  - insert custom routines
  - **This would be a good place to put extensions.**



C:\GitHub\engr-MCSB\lectures\Lecture2_source\PhysiCell\main.cpp - Notepad++

```cpp
                PhysiCell_globals.full_output_index++;
                PhysiCell_globals.next_full_save_time += PhysiCell_settings.full_save_interval;
            }

            // save SVG plot if it's time
            if( fabs( PhysiCell_globals.current_time - PhysiCell_globals.next_SVG_save_time  ) < 0.01 * diffusion_dt )
            {
                if( PhysiCell_settings.enable_SVG_saves == true )
                {
                    sprintf( filename , "%s/snapshot%08u.svg" , PhysiCell_settings.folder.c_str() , PhysiCell_globals.SVG_output_index );
                    SVG_plot( filename , microenvironment, 0.0 , PhysiCell_globals.current_time, cell_coloring_function );

                    PhysiCell_globals.SVG_output_index++;
                    PhysiCell_globals.next_SVG_save_time  += PhysiCell_settings.SVG_save_interval;
                }
            }

            // update the microenvironment
            microenvironment.simulate_diffusion_decay( diffusion_dt );

            // run PhysiCell
            ((Cell_Container *)microenvironment.agent_container)->update_all_cells( PhysiCell_globals.current_time );

            /*
              Custom add-ons could potentially go here.
            */

            PhysiCell_globals.current_time += diffusion_dt;
        }

        if( PhysiCell_settings.enable_legacy_saves == true )
        {
            log_output(PhysiCell_globals.current_time, PhysiCell_globals.full_output_index, microenvironment, report_file);
            report_file.close();
        }
    }
    catch( const std::exception& e )
```

C++ source file   length : 10,848   lines : 241   Ln : 157   Col : 95   Sel : 0 | 0   Windows (CR LF)   UTF-8   INS

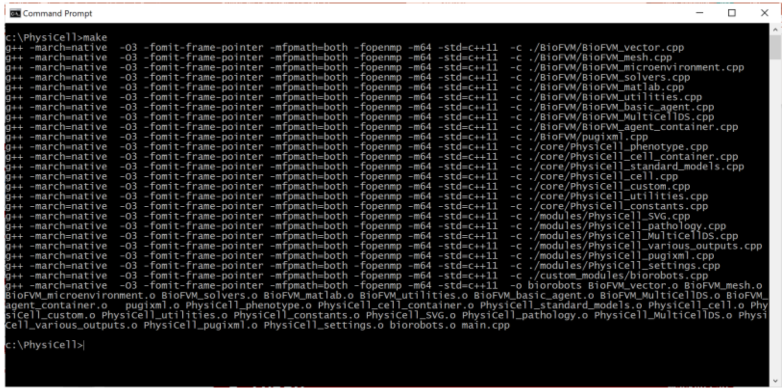
Now, let's get back to working with sample projects.

PhysiCell Project Essentials (2)

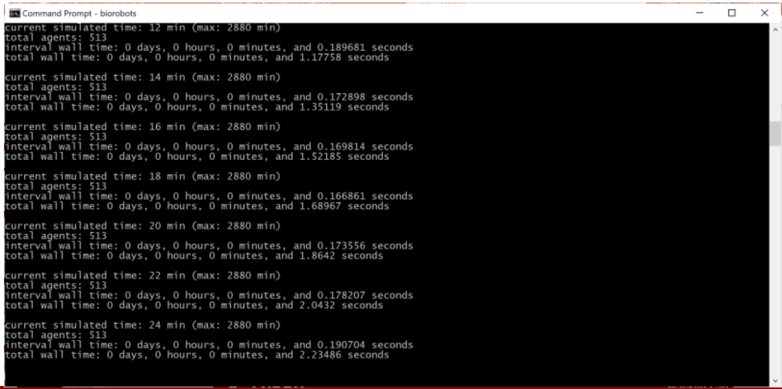
- Now, **compile the project**
  - `make`

- Then, **run the project**
  - `./biorobots` (Linux, MacOS)
  - `biorobots.exe` (Windows)

- This should take about 5 minutes


Command Prompt

```
c:\PhysiCell>make
g++ -march=native  -O3 -fomit-frame-pointer -mfpmath=both -fopenmp -m64 -std=c++11  -c ./BioFVM/BioFVM_vector.cpp
g++ -march=native  -O3 -fomit-frame-pointer -mfpmath=both -fopenmp -m64 -std=c++11  -c ./BioFVM/BioFVM_mesh.cpp
g++ -march=native  -O3 -fomit-frame-pointer -mfpmath=both -fopenmp -m64 -std=c++11  -c ./BioFVM/BioFVM_microenvironment.cpp
g++ -march=native  -O3 -fomit-frame-pointer -mfpmath=both -fopenmp -m64 -std=c++11  -c ./BioFVM/BioFVM_solvers.cpp
g++ -march=native  -O3 -fomit-frame-pointer -mfpmath=both -fopenmp -m64 -std=c++11  -c ./BioFVM/BioFVM_matlab.cpp
g++ -march=native  -O3 -fomit-frame-pointer -mfpmath=both -fopenmp -m64 -std=c++11  -c ./BioFVM/BioFVM_utilities.cpp
g++ -march=native  -O3 -fomit-frame-pointer -mfpmath=both -fopenmp -m64 -std=c++11  -c ./BioFVM/BioFVM_basic_agent.cpp
g++ -march=native  -O3 -fomit-frame-pointer -mfpmath=both -fopenmp -m64 -std=c++11  -c ./BioFVM/BioFVM_MultiCellDS.cpp
g++ -march=native  -O3 -fomit-frame-pointer -mfpmath=both -fopenmp -m64 -std=c++11  -c ./BioFVM/BioFVM_agent_container.cpp
g++ -march=native  -O3 -fomit-frame-pointer -mfpmath=both -fopenmp -m64 -std=c++11  -c ./BioFVM/pugixml.cpp
g++ -march=native  -O3 -fomit-frame-pointer -mfpmath=both -fopenmp -m64 -std=c++11  -c ./core/PhysiCell_phenotype.cpp
g++ -march=native  -O3 -fomit-frame-pointer -mfpmath=both -fopenmp -m64 -std=c++11  -c ./core/PhysiCell_cell_container.cpp
g++ -march=native  -O3 -fomit-frame-pointer -mfpmath=both -fopenmp -m64 -std=c++11  -c ./core/PhysiCell_standard_models.cpp
g++ -march=native  -O3 -fomit-frame-pointer -mfpmath=both -fopenmp -m64 -std=c++11  -c ./core/PhysiCell_cell.cpp
g++ -march=native  -O3 -fomit-frame-pointer -mfpmath=both -fopenmp -m64 -std=c++11  -c ./core/PhysiCell_custom.cpp
g++ -march=native  -O3 -fomit-frame-pointer -mfpmath=both -fopenmp -m64 -std=c++11  -c ./core/PhysiCell_utilities.cpp
g++ -march=native  -O3 -fomit-frame-pointer -mfpmath=both -fopenmp -m64 -std=c++11  -c ./core/PhysiCell_constants.cpp
g++ -march=native  -O3 -fomit-frame-pointer -mfpmath=both -fopenmp -m64 -std=c++11  -c ./modules/PhysiCell_SVG.cpp
g++ -march=native  -O3 -fomit-frame-pointer -mfpmath=both -fopenmp -m64 -std=c++11  -c ./modules/PhysiCell_pathology.cpp
g++ -march=native  -O3 -fomit-frame-pointer -mfpmath=both -fopenmp -m64 -std=c++11  -c ./modules/PhysiCell_MultiCellDS.cpp
g++ -march=native  -O3 -fomit-frame-pointer -mfpmath=both -fopenmp -m64 -std=c++11  -c ./modules/PhysiCell_various_outputs.cpp
g++ -march=native  -O3 -fomit-frame-pointer -mfpmath=both -fopenmp -m64 -std=c++11  -c ./modules/PhysiCell_pugixml.cpp
g++ -march=native  -O3 -fomit-frame-pointer -mfpmath=both -fopenmp -m64 -std=c++11  -c ./modules/PhysiCell_settings.cpp
g++ -march=native  -O3 -fomit-frame-pointer -mfpmath=both -fopenmp -m64 -std=c++11  -c ./custom_modules/biorobots.cpp
g++ -march=native  -O3 -fomit-frame-pointer -mfpmath=both -fopenmp -m64 -std=c++11  -o biorobots BioFVM_vector.o BioFVM_mesh.o
BioFVM_microenvironment.o BioFVM_solvers.o BioFVM_matlab.o BioFVM_utilities.o BioFVM_basic_agent.o BioFVM_MultiCellDS.o BioFVM_
agent_container.o   pugixml.o PhysiCell_phenotype.o PhysiCell_cell_container.o PhysiCell_standard_models.o PhysiCell_cell.o Phy
siCell_custom.o PhysiCell_utilities.o PhysiCell_constants.o PhysiCell_SVG.o PhysiCell_pathology.o PhysiCell_MultiCellDS.o Physi
Cell_various_outputs.o PhysiCell_pugixml.o PhysiCell_settings.o biorobots.o main.cpp

c:\PhysiCell>
```


Command Prompt - biorobots

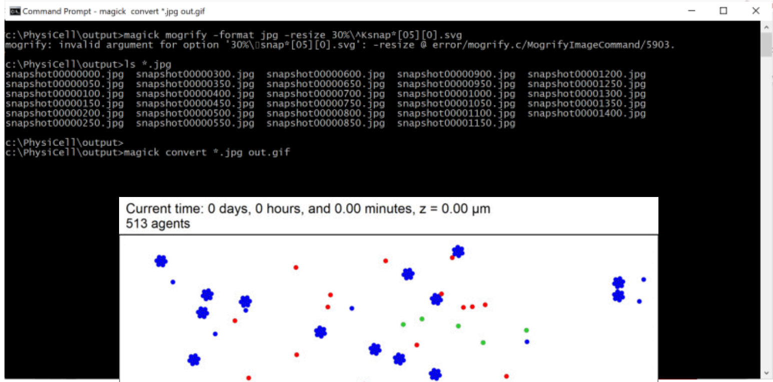
```
current simulated time: 12 min (max: 2880 min)
total agents: 513
interval wall time: 0 days, 0 hours, 0 minutes, and 0.189681 seconds
total wall time: 0 days, 0 hours, 0 minutes, and 1.17758 seconds

current simulated time: 14 min (max: 2880 min)
total agents: 513
interval wall time: 0 days, 0 hours, 0 minutes, and 0.172898 seconds
total wall time: 0 days, 0 hours, 0 minutes, and 1.35119 seconds

current simulated time: 16 min (max: 2880 min)
total agents: 513
interval wall time: 0 days, 0 hours, 0 minutes, and 0.169814 seconds
total wall time: 0 days, 0 hours, 0 minutes, and 1.52185 seconds

current simulated time: 18 min (max: 2880 min)
total agents: 513
interval wall time: 0 days, 0 hours, 0 minutes, and 0.166861 seconds
total wall time: 0 days, 0 hours, 0 minutes, and 1.68967 seconds

current simulated time: 20 min (max: 2880 min)
total agents: 513
interval wall time: 0 days, 0 hours, 0 minutes, and 0.173556 seconds
total wall time: 0 days, 0 hours, 0 minutes, and 1.8642 seconds

current simulated time: 22 min (max: 2880 min)
total agents: 513
interval wall time: 0 days, 0 hours, 0 minutes, and 0.178207 seconds
total wall time: 0 days, 0 hours, 0 minutes, and 2.0432 seconds

current simulated time: 24 min (max: 2880 min)
total agents: 513
interval wall time: 0 days, 0 hours, 0 minutes, and 0.190704 seconds
total wall time: 0 days, 0 hours, 0 minutes, and 2.23486 seconds
```

PhysiCell Project Essentials (3)

- **Look at saved data**
  - Most projects save data to ./output
    - ♦ XML files give metadata, mesh, and substrate info
    - ♦ MAT file save (compressed) substrate and cell data
    - ♦ SVG files are visual quick snapshots
    - ♦ More on loading XML / MAT files in Python later

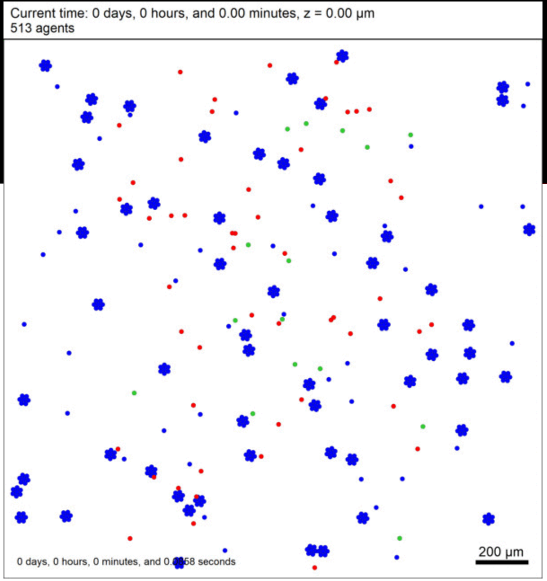
- Let's convert SVG to rescaled JPEG
  - `magick mogrify -format jpg -resize 30% snap*.svg`
    - ♦ Convert snapshot00000000.svg, snapshot00000001.svg, ...
  - `magick mogrify -format jpg -resize 30% snap*[05][0].svg`
    - ♦ Convert snapshot00000000.svg, snapshot00000050.svg, ...

- Now, let's create an animated GIF
  - `magick convert *.jpg out.gif`



Command Prompt - magick convert *.jpg out.gif

```
c:\PhysiCell\output>magick mogrify -format jpg -resize 30%\^Ksnap*[05][0].svg
mogrify: invalid argument for option '30%\snap*[05][0].svg': -resize @ error/mogrify.c/MogrifyImageCommand/5903.

c:\PhysiCell\output>ls *.jpg
snapshot00000000.jpg  snapshot00000300.jpg  snapshot00000600.jpg  snapshot00000900.jpg  snapshot00001200.jpg
snapshot00000050.jpg  snapshot00000350.jpg  snapshot00000650.jpg  snapshot00000950.jpg  snapshot00001250.jpg
snapshot00000100.jpg  snapshot00000400.jpg  snapshot00000700.jpg  snapshot00001000.jpg  snapshot00001300.jpg
snapshot00000150.jpg  snapshot00000450.jpg  snapshot00000750.jpg  snapshot00001050.jpg  snapshot00001350.jpg
snapshot00000200.jpg  snapshot00000500.jpg  snapshot00000800.jpg  snapshot00001100.jpg  snapshot00001400.jpg
snapshot00000250.jpg  snapshot00000550.jpg  snapshot00000850.jpg  snapshot00001150.jpg

c:\PhysiCell\output>
c:\PhysiCell\output>magick convert *.jpg out.gif
```



Working with the images

- To convert all the SVG files to PNG format

```
magick mogrify -format png snap*.svg
```

- To convert every SVG file ending in 0 or 5 to JPG format

```
magick mogrify -format jpg snap*[05].svg
```

- To convert the JPG files to an animated GIF

```
magick convert *.jpg out.gif
```

- To create an mp4 movie:

```
ffmpeg -r 24 -f image2 -i snapshot%08d.jpg -vcodec libx264 -pix_fmt yuv420p -strict -2 -tune
animation -crf 15 -acodec aac out.mp4
```

Handy tricks!

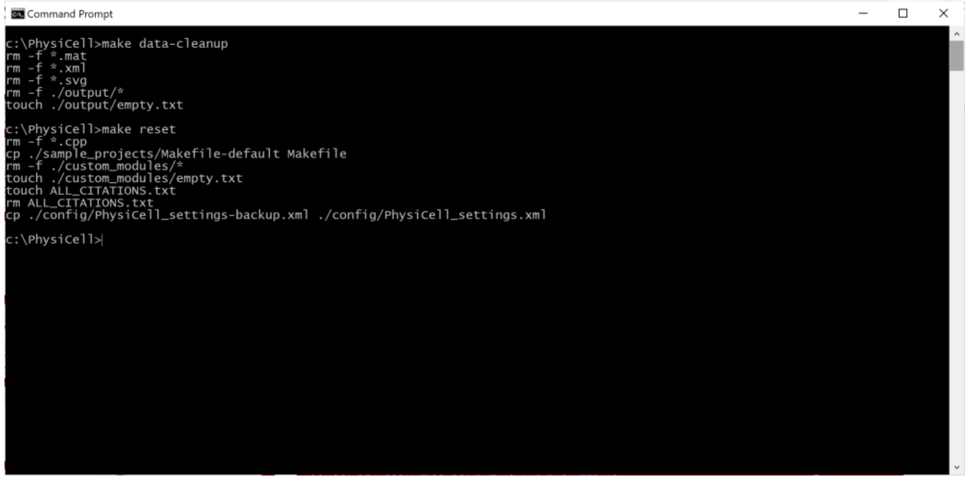
Use make jpeg to create a full set of JPGs

Use make movie easily create the mp4.

PhysiCell Project Essentials (4)

- **Data cleanup**
  - Clean up data to get ready for another run
  - `make data-cleanup`

- **Reset** to a clean slate
  - De-populate the project
  - Get ready for another project
  - `make reset`



```
c:\PhysiCell>make data-cleanup
rm -f *.mat
rm -f *.xml
rm -f *.svg
rm -f ./output/*
touch ./output/empty.txt

c:\PhysiCell>make reset
rm -f *.cpp
cp ./sample_projects/Makefile-default Makefile
rm -f ./custom_modules/*
touch ./custom_modules/empty.txt
touch ALL_CITATIONS.txt
rm ALL_CITATIONS.txt
cp ./config/PhysiCell_settings-backup.xml ./config/PhysiCell_settings.xml

c:\PhysiCell>
```

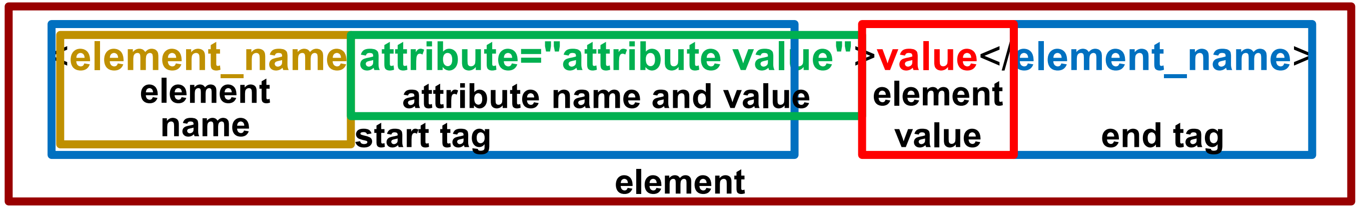
Changing settings in a project

XML Refresher (1)

- XML stands for e**X**tensible **M**arkup **L**anguage
  - (Think of it as a generalization of HTML.)

- Information in XML are stored in elements. Key elements are:
  - element name in a start tag
  - attributes and values
  - element value

- If an element has a value, it must have a matching end tag:



- If an element has attributes but no value, you can use a more compact form:

```
<element_name attribute1="attribute 1 value" attribute2="attribute 2 value" />
```

XML Refresher (2)

- Just like HTML, XML can have sub-elements:

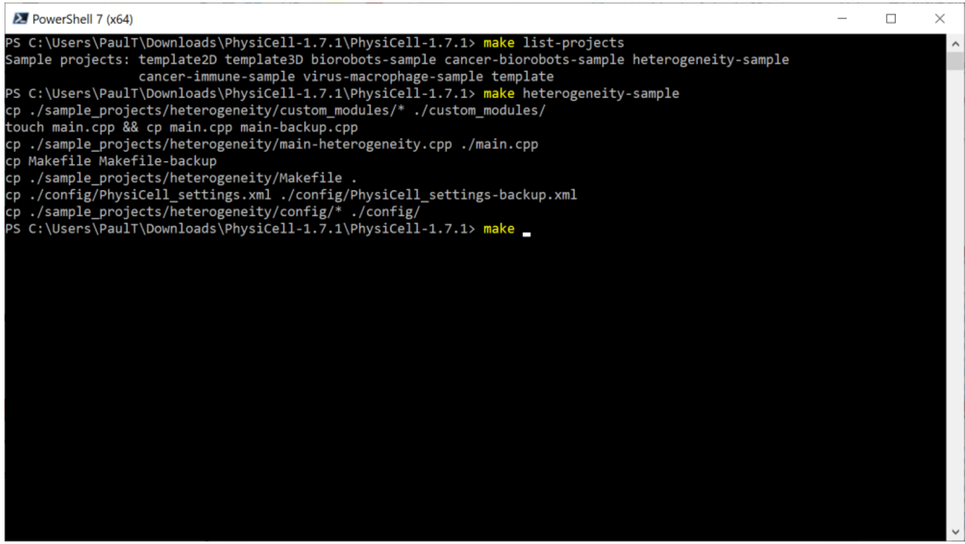
```
<element_name attribute1="attribute 1 value" attribute2="attribute 2 value">
       <subelement_name attribute="attribute value">subvalue1</subelement_name >
       <subelement_name attribute="attribute value">subvalue2</subelement_name >
</element_name >
```

- By convention:
  - the name of the element is a parameter name
  - the element's value is the parameter value
  - attributes are used to store metadata or other clarifications (e.g., units)

```
<diffusion_coefficient units="micron/min^2">1000</diffusion_coefficient>
```

First, populate the cancer heterogeneity project

- List all available sample projects
- Populate the cancer heterogeneity project
- Build the project
- Change some settings (next slide)



```
PS C:\Users\PaulT\Downloads\PhysiCell-1.7.1\PhysiCell-1.7.1> make list-projects
Sample projects: template2D template3D biorobots-sample cancer-biorobots-sample heterogeneity-sample
                 cancer-immune-sample virus-macrophage-sample template
PS C:\Users\PaulT\Downloads\PhysiCell-1.7.1\PhysiCell-1.7.1> make heterogeneity-sample
cp ./sample_projects/heterogeneity/custom_modules/* ./custom_modules/
touch main.cpp && cp main.cpp main-backup.cpp
cp ./sample_projects/heterogeneity/main-heterogeneity.cpp ./main.cpp
cp Makefile Makefile-backup
cp ./sample_projects/heterogeneity/Makefile .
cp ./config/PhysiCell_settings.xml ./config/PhysiCell_settings-backup.xml
cp ./sample_projects/heterogeneity/config/* ./config/
PS C:\Users\PaulT\Downloads\PhysiCell-1.7.1\PhysiCell-1.7.1> make _
```

How to change settings in XML

- Open config/PhysiCell_settings.xml
- Major sections:
  - **domain** -- how big of a region to simulate
  - **overall** -- how long to simulate, time step sizes
  - **parallel** -- OpenMP settings
  - **save** -- how often to save SVG images and full data
  - **microenvironment** -- settings on diffusing substrates
  - **user_parameters** -- model-specific settings
  - **cell_definitions** -- set baseline cell properties

Exercise: change settings and run

- Let's set the maximum simulation time to 2160 minutes
- Let's set the domain to [-500,500] x [-500,500] to speed it up
- Let's set the oncoprotein standard deviation to 3
- Let's set the max oncoprotein to 9 (3 standard deviations)
- Compile and run as before.

Let's set options and run (1)

- Open `./config/PhysiCell-settings.xml`
- Let's set the domain size in the **domain** block
  - Switch to [-500,500] x [-500,500] x [-10,10] to speed it up

```
<PhysiCell_settings version="devel-version">
        <domain>
                <x_min>-500</x_min>
                <x_max>500</x_max>
                <y_min>-500</y_min>
                <y_max>500</y_max>
                <z_min>-10</z_min>
                <z_max>10</z_max>
                <dx>20</dx>
                <dy>20</dy>
                <dz>20</dz>
                <use_2D>true</use_2D>
        </domain>
```

Let's set options and run (2)

- Let's also look at the **user_parameters** block
  - Let's change the oncoprotein standard deviation (`oncoprotein_sd`) to 3 (more variation)
  - Let's change the max oncoprotein (`oncoprotein_max`) to mean + 3 sds = 1 + 9 = 10

```
<user_parameters>
  <tumor_radius type="double" units="micron">250.0</tumor_radius>
  <oncoprotein_mean type="double" units="dimensionless">
    1.0</oncoprotein_mean>
  <oncoprotein_sd type="double" units="dimensionless">3.0</oncoprotein_sd>
  <oncoprotein_min type="double" units="dimensionless">0.0</oncoprotein_min>
  <oncoprotein_max type="double" units="dimensionless">10</oncoprotein_max>
  <random_seed type="int" units="dimensionless">0</random_seed>
</user_parameters>
```

Let's set options and run (3)

- Let's look at the **overall** block
  - Set max time to 1.5 days = 1.5 x 24 x 60 = 2160 minutes

```
<overall>
        <max_time units="min">2160</max_time> <!-- 36 h * 60 min -->
        <time_units>min</time_units>
        <space_units>micron</space_units>
```

- Let's look at the **save** block
  - Set the full save interval to 6 hours = 360 minutes

```
<save>
        <folder>output</folder> <!-- use . for root -->
        <full_data>
                <interval units="min">360</interval>
                <enable>true</enable>
        </full_data>
```

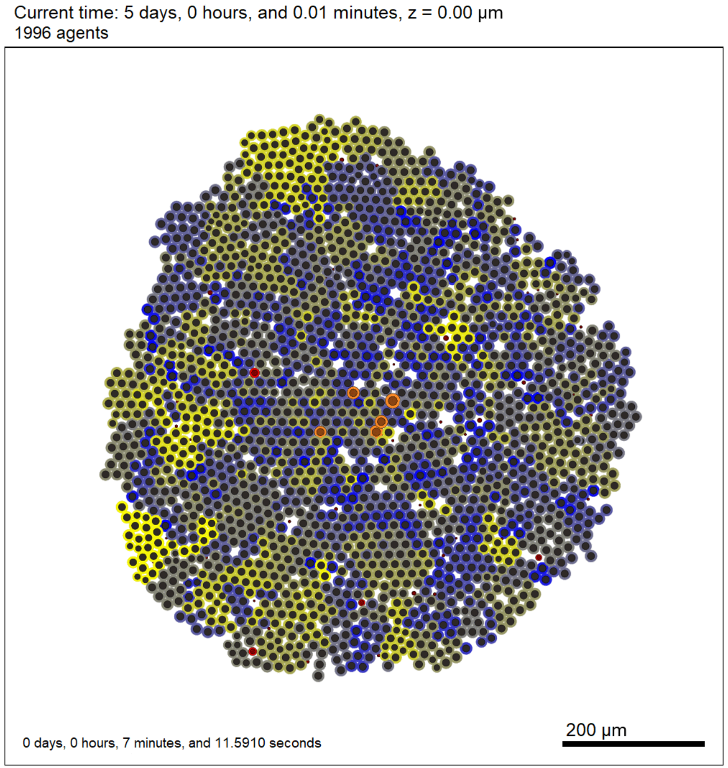
- Now, run! (`./heterogeneity`)

Let's do a quick visualization

- `magick mogrify -format jpg *.svg`
- `magick convert *.svg out.gif`

- We can see that the yellow cells eventually "win": they grow faster and form microcolonies within the tumor

- The effect is greatest on the outside edge: They have access to more O_2 here



Funding Acknowledgements

PhysiCell Development:

- Breast Cancer Research Foundation
- Jayne Koskinas Ted Giovanis Foundation for Health and Policy
- National Cancer Institute (U01CA232137)
- National Science Foundation (1720625)

Training Materials:

- Administrative supplement to NCI U01CA232137 (Year 2)