WormGUIDES Desktop JavaFX Application

Code README

Purpose:

To provide a high level view of the logic of the package organization.

To detail the functionality in each of the major classes.

To detail how communication between components works.

Package Organization

The top level of the WormGUIDES directory is divided into the following components:

* *documentation/* contains **code leve**l, **Notes**, **URLs** and **Window3D parameters** documentation
* *lib/* stores external libraries that WormGUIDES leverages for functionality
  + *jmf.jar* – Java Media Framework JAR for taking screenshots and videos
* LICENCE – The GNU General Public License
* *META-INF/* contains the MANIFEST.MF file that IntelliJ uses (our suggested IDE for the WormGUIDES developer community)
* *out/* contains compiled source files
* README.md
* *src/* contains the source files, data files, and configuration files for WormGUIDES (descripted in detail below)
* WormGUIDES.iml– IntelliJ configuration file
* WormGUIDES.jar – the WormGUIDES executable file, of type Java Archive

The *src/* directory’s organizing principle is files that are specific to the WormGUIDES application and those that exist independently that are leveraged by the application. All files internal to the application are stored in the *wormguides/* directory and will be explained below. There are four directories external storing files external to WormGUIDES. They are:

* *acetree/* contains the lineage data, the interface for accessing this data, and the loader and database which implements the interface
  + LineageData.java – the data structure interface for querying the lineage data
  + *nucleifiles/* contains one file per time point during embryogenesis which defines cell positions
  + *tablelineagedata/* 
    - AceTreeTableLineageDataLoader.java – The loader that processes the nuclei files and places them in the TableLineageData
    - TableLineageData.java – Table that keeps the lineage data in **Frame** data structures
* *connectome/* contains the data file and classes for defining the querying the connectome
  + Connectome.java
  + ConnectomeLoader.java
  + NeuronalSynapse.java
  + NeuronConnect.csv
  + SynapseType.java
* *partslist/* contains the data files for the partslist and its accompanying class, with a subdirectory that contains a data file defining cells deaths and its accompanying class
  + *celldeaths/*
    - CellDeaths.csv
    - CellDeaths.java
  + PartsList.java
  + partslist.txt
* *search/* files used for searching the model and querying WormBase
  + SearchType.java
  + SearchUtil.java
  + WormBaseQuery.java

The *wormguides/* directory’s organizing principle follows the architecture of the WormGUIDES application. WormGUIDES is an MVC application (Model, View, Controller) which provides users with a graphical view of the model, and controllers that facilitate model manipulation that trigger view updates. The three main directories within *wormguides/* are:

* *controllers/* contains controllers classes that correspond to graphical controllers defined in view, and other controllers for functionality
  + ContextMenuController.java
  + InfoWindowLinkController.java
  + RootLayoutController.java
  + RotationController.java
  + RuleEditorController.java
  + StoryEditorController.java
  + Window3DController.java
* *models/* defines the models that form the backend of the application, in addition to the external lineage data
  + *analogouscell/*
  + *anatomy/*
  + *camerageometry/*
  + *cellcase/*
  + *colorrule/*
  + LineageTree.java
  + *obj\_files/*
  + *production\_info\_file/*
  + *shapes\_file/*
  + *subscenegeometry/*
* *view/* defines the views that present the underlying model
  + DraggableTab.java
  + *graphicalrepresentations/*
  + *icons/*
  + *infowindow/*
  + *layouts/*
  + *popups/*
  + *urlwindow/*

The other files and directories that define the WormGUIDES application within *wormguides/* are:

* *layers/* defines the functionality of the **Display**, **Search**, **Stories**, and **Structures** tabs on the RootLayout
* *loaders/* contains classes for loading elements for the 3D window and model parameters
  + GeometryLoader.java
  + IconImageLoader.java
  + NoteImageLoader.java
  + ParametersLoader.java
  + ProductionInfoLoader.javas
* MainApp.java
* *resources/*
* *stories/*
* *util/*

Code Level Documentation of Major Classes