

# What is Possible in BRIDGES?

Kalpathi Subramanian<sup>1</sup>, Erik Saule<sup>1</sup>, Jamie Payton<sup>2</sup>  
krs@uncc.edu, esaule@uncc.edu, payton@temple.edu

<sup>1</sup>The University of North Carolina at Charlotte

<sup>2</sup>Temple University

BRIDGES Workshop, June 24-26, 2024

# Table of Contents

1 Datasets

2 Visualizations

3 Assignments

## IGN Games Dataset

This dataset contains a collection of game reviews at IGN and is adapted from Reddit User: CDanger (Twitter: @coreyaustin) who scraped the data from IGN's website.

Data end point: [IGN Game Link](#)

The program snippet below illustrates how to access the IGN data.

## Accessing IGN Games Data: An Example BRIDGES program

[Java](#)   [C++](#)   [Python](#)

```
import java.util.List;
import java.util.Random;
import bridges.connect.Bridges;
import bridges.connect.DataSource;
import bridges.data_src_dependent.Game;

// this program fragment illustrates how to access the IGN Games data
public class game_snippet {
    public static void main(String[] args) throws Exception {

        //create the Bridges object
        Bridges bridges = new Bridges(YOUR_ASSIGNMENT_NUMBER, "YOUR_USER_ID", "YOUR_APP_KEY");

        /* Get a List of Game objects from Bridges */
        DataSource ds = bridges.getDataSource();
        List<Game> mylist = ds.getGameData();

        // Inspect a random Game object
        Game game1 = mylist.get(new Random().nextInt(mylist.size()));
        System.out.println(game1.getTitle());
        System.out.println(game1.getPlatformType());
        System.out.println(game1.getRating());
        System.out.println(game1.getGenre());

    }
}
```

- Copy/paste the code into your environment and compile.
- Assuming all your code is correct and it compiles correctly, the code will print a record of the data to the console.

## bridges.data\_src\_dependent.Game Class Reference

[Public Member Functions](#) | [Package Attributes](#) | [List of all members](#)

### Detailed Description

A Game object, used along with the Games data source.

This is a convenience class provided for users who wish to use this data source as part of their application. It provides an API that makes it easy to access the attributes of this data set.

One would not normally create an object of this type, but rather obtain them through a call to `bridges::connect::DataSource::getGameData()`.

Each game has a title, platform on which it can be played, rating and a list of genres

Refer to tutorial examples on how to use this dataset: [https://bridgesuncc.github.io/tutorials/Data IGN\\_Games.html](https://bridgesuncc.github.io/tutorials/Data IGN_Games.html)

#### Author

Kalpathi Subramanian

#### Date

2/17, 12/26/20

### Public Member Functions

<code>Game ()</code>
<code>Game (String title, String platform, double rating, Vector&lt; String &gt; genre)</code>
<code>String getTitle ()</code>
<code>void setTitle (String title)</code>
<code>String getPlatformType ()</code>
<code>void setPlatformType (String platform)</code>
<code>double getRating ()</code>
<code>void setRating (double rating)</code>
<code>Vector&lt; String &gt; getGenre ()</code>
<code>void setGenre (Vector&lt; String &gt; genre)</code>

### Package Attributes

<code>String platform</code>
<code>Vector&lt; String &gt; genre</code>

### Constructor & Destructor Documentation



## USGS Earthquake Data

This data set is extracted as Tweets from the [US Geological Survey \(USGS\)](#). The BRIDGES API extracts the quake data records periodically and retains the most recent 5000 quakes in a database. These are accessed by the BRIDGES client, reformatted as a list of objects for use in assignments.

The program snippet below illustrates how to access the earthquake data.

### Accessing Earthquake Data: An Example BRIDGES program

[Java](#) [C++](#) [Python](#)

```
import java.util.List;
import bridges.connect.Bridges;
import bridges.connect.DataSource;
import bridges.data_src_dependent.EarthquakeUSGS;

// This program fragment illustrates how to access the USGS earthquake data
public class eq_snippet {
    public static void main(String[] args) throws Exception {

        // create Bridges object
        Bridges bridges (YOUR_ASSIGNMENT_NUMBER, "YOUR_USER_ID", "YOUR_API_KEY");
        // read the earth quake data and build the BST
        bridges.setTitle("Accessing USGS Earthquake Data (USGS Data)");

        DataSource ds = bridges.getDataSource();

        // get 100 most recent earthquake records
        List<EarthquakeUSGS> eq_list = ds.getEarthquakeUSGSData(100);

        // print the first record
        System.out.println("Earthquake 0:");
        System.out.println("Magnitude:" + eq_list.get(0).getMagnitude() + "\n"
            + "Date:" + eq_list.get(0).getTime() + "\n"
            + "Location:" + eq_list.get(0).getLocation() + "\n"
            + "Lat/Long:" + eq_list.get(0).getLatit() + ","
            + eq_list.get(0).getLongit());
    }
}
```



## Detailed Description

This class stores the information of an earthquake, retrieved from US Geological Survey Data tweet source.

One would normally not create an object of that type but rather obtain one by calling `bridges::connect::DataSource::getEarthquakeUSGSData()`

Refer to the tutorial on how to use this dataset: [https://bridgesuncc.github.io/tutorials/Data\\_EQ\\_USGS.html](https://bridgesuncc.github.io/tutorials/Data_EQ_USGS.html)

### Author

Mihai Mehdint, Kalpathi Subramanian

### Date

12/26/20

Modifications: removed dependencies on Tweet and its ancestors

## Public Member Functions

```
EarthquakeUSGS ()
EarthquakeUSGS (String content, Date date2, double magnitude, double longit, double latit, String location, String title, String url, String properties)
String getProperties ()
void setProperties (String properties)
double getLatit ()
void setLatit (double latit)
double getLongit ()
void setLongit (double longit)
String getLocation ()
void setLocation (String location)
String getTitle ()
void setTitle (String title)
String getUrl ()
void setUrl (String url)
void setMagnitude (double magnitude)
void setTime (String t)
String getTime ()
EarthquakeUSGS (EarthquakeUSGS eq)
void eqProperties (String prop)
void setMagnitude (Double mag)
String enterCarriageReturn (String str)
double getMagnitude ()
```

Bridges - Elevation Tutor x +

[bridgesuncc.github.io/tutorials/Data\\_Elevation.html](https://bridgesuncc.github.io/tutorials/Data_Elevation.html)

### Step 1: Suggested Examples:

```
data_source.get_elevation_data([36.8241, -116.8369, 35.7797, -117.4074])
data_source.get_elevation_data([49.0872, -113.6184, 47.3746, -114.8143])
data_source.get_elevation_data([40.1924, -79.3980, 37.5021, -81.8810])
```

## Accessing Elevation Data: An Example BRIDGES program

Java C++ Python

```
import java.util.List;
import java.util.Random;
import bridges.connect.Bridges;
import bridges.connect.DataSource;
import bridges.data_src_dependent.ElevationData;

// a program fragment to access the Elevation data
public class elevation_snippet {
    public static void main(String[] args) throws Exception {

        //create the Bridges object
        Bridges bridges = new Bridges(YOUR_ASSIGNMENT_NUMBER, "YOUR_USER_ID", "YOUR_API_KEY");

        // Get a List of Elevation data records from Bridges
        DataSource ds = bridges.getDataSource();
        ElevationData elev_data = ds.getElevationData(41.33133177632377,
            -98.02593749997456, 42.508577297430456, -96.94531249997696, 0.02);

        System.out.println("Width: " + elev_data.getCols());
        System.out.println("Height: " + elev_data.getRows());
        System.out.println("Resolution: " + elev_data.getCellSize());
        System.out.println("Lower Left Corner: " + elev_data.getllx() + " " + elev_data.getyll());

        System.out.println("first 10 elevations:");
        for (int i = 0; i < 10; ++i) {
            System.out.println(elev_data.getData()[0][i]);
        }
    }
}
```

Bridges-Java: bridges.data\_ x +

[https://bridgesuncc.github.io/doc/java-api/current/html/bridges/data\\_src\\_dependent/ElevationData.html](https://bridgesuncc.github.io/doc/java-api/current/html/bridges/data_src_dependent/ElevationData.html)

Public Member Functions | List of all members

## bridges.data\_src\_dependent.ElevationData Class Reference

### Detailed Description

Object that holds elevation data retrieved from NOAA repository.

A user would not normally create an `ElevationData` object but rather obtain one from calling `bridges::connect::DataSource::getElevationData()`

A tutorial on how to use the Elevation dataset is available at: [https://bridgesuncc.github.io/tutorials/Data\\_Elevation.html](https://bridgesuncc.github.io/tutorials/Data_Elevation.html)

### Author

Jay Strahler

### Date

12/26/20

### Public Member Functions

<code>ElevationData ()</code>
<code>ElevationData (int[] data, int cols, int rows, double xll, double yll, double cellsize, int maxVal, int minVal)</code>
<code>int[] getData ()</code>
<code>int getVal (int r, int c)</code>
<code>void setData (int[] data)</code>
<code>int getCols ()</code>
<code>void setCols (int cols)</code>
<code>int getRows ()</code>
<code>void setRows (int rows)</code>
<code>double getllx ()</code>
<code>void setllx (double xll)</code>
<code>double getyll ()</code>
<code>void setyll (double yll)</code>
<code>double getMaxVal ()</code>
<code>void setMaxVal (int maxVal)</code>
<code>int getMinVal ()</code>
<code>void setMinVal (int minVal)</code>

## Bridges - Lyrics Tutorial

```
ds.get_song("Enter Sandman", "Metallica")
ds.get_song("Around the World", "Daft Punk")
ds.get_song("Never Gonna Give You Up", "Rick Astley")
```

## Accessing Song Lyrics Data: An Example BRIDGES program

Java

C++

Python

```
import java.util.List;
import java.util.Random;
import bridges.connect.Bridges;
import bridges.connect.DataSource;
import bridges.data_src_dependent.Song;
import java.util.ArrayList;

// a program fragment illustrating how to access the song lyrics data set

public class song_lyrics_snippet {
    public static void main(String[] args) throws Exception {

        //create the Bridges object
        Bridges bridges = new Bridges(YOUR_ASSIGNMENT_NUMBER, "YOUR_USER_ID", "YOUR_API_KEY");

        // set title
        bridges.setTitle("Accessing_Song_Data");

        // create data source object
        DataSource ds = bridges.getDataSource();

        // Get Song data
        Song s = ds.getSong("Harder Faster Better Stronger", "Daft Punk");

        // print song attributes
        System.out.println(s.getSongTitle());
        System.out.println(s.getArtist());
        System.out.println(s.getAlbumTitle());
        System.out.println(s.getReleaseDate());
        System.out.println(s.getLyrics());

    }
}
```

## Bridges-Java: bridges.data\_src\_

[Main Page](https://bridgesuncc.github.io/doc/java-api/current/html/index.html) [Packages](#) [Classes](#) [Files](#)

 Search
[bridges](#) [data\\_src\\_dependent](#) [Song](#)
[Public Member Functions](#) | [Package Attributes](#) | [List of all members](#)
**bridges.data\_src\_dependent.Song Class Reference****Detailed Description**

A [Song](#) object, used along with the Songs data source (using the Genius API).

This is a convenience class provided for users who wish to use this data source as part of their application. It provides an API that makes it easy to access the attributes of this data set. The [Song](#) object is typically obtained from calling [bridges::connect::DataSource::getSongData\(\)](#) or [bridges::connect::DataSource::getSong\(\)](#).

Refer to tutorial for example of using this feature: [https://bridgesuncc.github.io/tutorials/Data\\_Song\\_Lyrics.html](https://bridgesuncc.github.io/tutorials/Data_Song_Lyrics.html)

**Author**

David Burlinson

**Date**

5/21/18

**Public Member Functions**

<a href="#">Song ()</a>	<a href="#">Song (String artist, String song, String album, String lyrics, String release_date)</a>
<a href="#">String getArtist ()</a>	<a href="#">void setArtist (String artist)</a>
<a href="#">String getSongTitle ()</a>	<a href="#">void setSongTitle (String song)</a>
<a href="#">String getAlbumTitle ()</a>	<a href="#">void setAlbumTitle (String album)</a>
<a href="#">String getLyrics ()</a>	<a href="#">void setLyrics (String lyrics)</a>
<a href="#">String getReleaseDate ()</a>	<a href="#">void setReleaseDate (String release_date)</a>

**Package Attributes**[String song](#)[String title](#)

**Step 1: Suggested Examples:**

```
data_source.get_gutenberg_book_metadata("Shakespeare, William", "author")
data_source.gutenberg_book_text(2701)
```

The program snippet below illustrates how to access this dataset.

## Accessing Gutenberg Data: An Example BRIDGES program

Java    C++    Python

```
import bridges.connect.Bridges;
import bridges.connect.DataSource;
import bridges.data_src_dependent.Amenities;
import bridges.data_src_dependent.*;
import java.util.*;

public class gutenberg_snippet {
    public static void main(String[] args) throws Exception {

        //create the Bridges object
        Bridges bridges = new Bridges("YOUR_USER_ID", "YOUR_APP_KEY");

        //create a bridges datasource object to use and
        //retrieve the gutenberg books
        DataSource ds = bridges.getDataSource();

        //Get Meta data for a single book

        //Get Meta data for a book

        System.out.println("Meta data for 1 book: Moby_Dick,_by_Id");
        GutenbergMeta b = ds.getAGutenbergBookMetaData(2701);
        System.out.println("\tId: " + b.getId());
        System.out.println("\tTitle: " + b.getTitle());
        System.out.println("\tLanguage: " + b.getLanguage());
        System.out.println("\tDate Added: " + b.getDate());

        System.out.println("Retrieving books by title: Pride_and_Prejudice");
        List<GutenbergMeta> blist = ds.getGutenbergBookMetaData("Pride_and_Prejudice", "title");
```

## bridges.data\_src\_dependent.GutenbergMeta Class Reference

### Detailed Description

Stores the meta data of a book as stored by project Gutenberg.

This object stores the meta data of a book such as its title, author and genre. The meta data come from <https://www.gutenberg.org/>.

Objects of this type are typically not constructed by the user but returned by a call to our Gutenberg API such as `bridges::connect::DataSource::getGutenbergBookMetaData()` or `bridges::connect::DataSource::getAGutenbergBookMetaData(int)`.

The object does not contain the text of the book itself. Though it can be obtained using `bridges::connect::DataSource::getGutenbergBookText()`

A tutorial of how to use the Gutenberg data in BRIDGES is available: [https://bridgesuncc.github.io/tutorials/Data\\_Gutenberg.html](https://bridgesuncc.github.io/tutorials/Data_Gutenberg.html)

### Public Member Functions

<code>GutenbergMeta ()</code>	
<code>GutenbergMeta (int id, String title, String lang, String date, String[] authors, String[] genres, String[] loc)</code>	
<code>int getId ()</code>	GutenbergID of the book. More...
<code>void setId (int id)</code>	
<code>String getTitle ()</code>	Title of the book. More...
<code>void setTitle (String title)</code>	
<code>String getLanguage ()</code>	language of the book. More...
<code>void setLanguage (String lang)</code>	
<code>String getDate ()</code>	date at which the book was added to project Gutenberg. More...
<code>void setDate (String date)</code>	
<code>String[] getAuthors ()</code>	List of authors. More...
<code>void setAuthors (String[] authors)</code>	
<code>String[] getGenres ()</code>	List of genres. More...
<code>void setGenres (String[] genres)</code>	
<code>String[] getLoc ()</code>	List of library of congress classifications. More...

- **BRIDGES Initialization, Credentials:** This is typically done **once** at the beginning of the application. Credentials include your user id and application authentication id (available via your BRIDGES login). Here this is required to access the DataSource object which controls all calls to external datasets.
- **Retrieving the data** This is a single call to a method in DataSource to return object(s) of that data type, and may include parameters.
- **Printing an example record** The program will print an example record of this dataset, to illustrate its attributes.

[Java](#)   [C++](#)   [Python](#)

```
import java.util.List;
import java.util.Random;
import bridges.connect.Bridges;
import bridges.connect.DataSource;
import bridges.data_src_dependent.Shakespeare;

// a program fragment to access the Shakespeare data (plays, poems)
public class shakespeare_snippet {
    public static void main(String[] args) throws Exception {

        //create the Bridges object
        Bridges bridges = new Bridges(YOUR_ASSIGNMENT_NUMBER, "YOUR_USER_ID", "YOUR_APP_KEY");

        // Get a List of Shakespeare objects from Bridges
        DataSource ds = bridges.getDataSource();
        List<Shakespeare> mylist = ds.getShakespeareData();

        // Inspect a random Shakespeare object
        Shakespeare work1 = mylist.get((new Random()).nextInt(mylist.size()));
        System.out.println(work1.getTitle());
        System.out.println(work1.getType());
        System.out.println(work1.getText().substring(0, Math.min(100, work1.getText().length())) + "...");

    }
}
```

- Copy/paste the code into your environment and compile.
- Assuming all your code is correct and it compiles correctly, the code will print a record of the data to the console.

## bridges.data\_src\_dependent.Shakespeare Class Reference

[Public Member Functions](#) | [Package Attributes](#) | [List of all members](#)

### Detailed Description

A Shakespeare Data source object containing sonnets, poems and plays.

This is a convenience class provided for users who wish to use this data source as part of their application. It provides an API that makes it easy to access the attributes of this data set.

One would not normally create an object of this type but rather get one by calling `bridges::connect::DataSource::getShakespeareData()`

Refer to the tutorial on how to use this dataset: [https://bridgesuncc.github.io/tutorials/Data\\_Shakespeare.html](https://bridgesuncc.github.io/tutorials/Data_Shakespeare.html)

#### Author

Kalpathi Subramanian

#### Date

2/1/17, 12/26/20

### Public Member Functions

`Shakespeare()`

`Shakespeare (String title, String type, String text)`

`String getTitle ()`

`void setTitle (String title)`

`String getType ()`

`void setType (String type)`

`String getText ()`

`void setText (String text)`

### Package Attributes

`String type`

`String text`

### Constructor & Destructor Documentation

◆ `Shakespeare()` [1/2]

## IMDB Actor/Movie Data

This data set is a curated subset of IMDB actor/movie dataset, with each record representing an actor name and a movie in which the actor appeared. There are 1814 actor/movie pairs in this dataset; this dataset is particularly suited to solving the 8 number problem, using graphs and Breadth First Search (BFS).

The program snippet below illustrates how to access this dataset.

## Accessing IMDB Data: An Example BRIDGES program

Java

C++

Python

```
import java.util.List;
import java.util.Random;
import bridges.connect.Bridges;
import bridges.connect.DataSource;
import bridges.data_src_dependent.ActorMovieIMDB;

// A test class that fetches a random movie from IMDB then sends the
//   actors name and movie title + release date to the console as output.
//   There is no visual output to the users BRIDGES gallery after running this code.

public class imbd_snippet {
    public static void main(String[] args) throws Exception {

        //create the Bridges object
        Bridges bridges = new Bridges(YOUR_ASSIGNMENT_NUMBER, "YOUR_USER_ID", "YOUR_APT_KEY");

        DataSource ds = bridges.getDataSource();

        // Get a List of ActorMovieIMDB objects from Bridges
        List<ActorMovieIMDB> mylist = ds.getActorMovieIMDBData(1813);

        // Inspect a random ActorMovieIMDB object
        ActorMovieIMDB pair1 = mylist.get(new Random().nextInt(mylist.size()));
        System.out.println(pair1.getActor());
        System.out.println(pair1.getMovie());

    }
}
```

- Copy/paste the code into your environment and compile.

## bridges.data\_src\_dependent.ActorMovieIMDB Class Reference

### Detailed Description

This class represents an actor-movie pair object, to be used with the IMDB actor-movie dataset.

A user typically does not create an `ActorMovieIMDB` object. They are usually returned by `bridges::connect::DataSource::getActorMovieIMDBData()` and `bridges::connect::DataSource::getActorMovieIMDBData2()`

Check out the tutorial on the IMDB dataset at [https://bridgesuncc.github.io/tutorials/Data\\_IMDB.html](https://bridgesuncc.github.io/tutorials/Data_IMDB.html)

### Author

Kalpathi Subramanian

### Public Member Functions

<code>ActorMovieIMDB ()</code>	
<code>String getActor ()</code>	get the actor from this object More...
<code>void setActor (String a)</code>	set the actor from this object More...
<code>String getMovie ()</code>	get the movie from this object More...
<code>void setMovie (String m)</code>	set the movie from this object More...
<code>double getMovieRating ()</code>	get the movie rating from this object More...
<code>void setMovieRating (double r)</code>	set the movie rating for this object More...
<code>Vector&lt; String &gt; getGenres ()</code>	get the genres of this movie More...
<code>void setGenres (Vector&lt; String &gt; g)</code>	set the genres for this movie More...

### Package Attributes

`String movie`

The program snippet below illustrates how to access the Wikidata actor/movie data using the Wikidata source.

Java

C++

Python

```
/*
 * Created by Lucas Estrella on 1/31/2017.
 * lestell@uncc.edu
 */
import java.util.ArrayList;

import bridges.connect.Bridges;
import bridges.connect.DataSource;
import bridges.data_src_dependent.ActorMovieWikidata;

// This program fragment illustrates how to access and read the Wikidata actor/movie data

public class wikidata_actor_snippet {

    public static void main(String[] args) throws Exception {

        // note: you must fill in with your Bridges credentials
        Bridges bridges = new Bridges(YOUR_ASSIGNMENT_NUMBER, "YOUR_USER_ID",
            "YOUR_API_KEY");
        // set title
        bridges.setTitle("Accessing Wikidata Movie/Actor Data");

        // create data source object
        DataSource ds = bridges.getDataSource();

        // Get a List of Wikidata actor/movie records from Bridges
        // get the actor movie Wikidata data through the BRIDGES API for 1955.
        // data are available from the early 20th century to now.
        ArrayList<ActorMovieWikidata> list = ds.getWikidataActorMovie(1955, 1955);

        System.out.println("Data Records in 1955: " + list.size() + "\n");

        // print out the first 3 records of the dataset
        for (int k = 0; k < 3; k++) {
            System.out.println("Actor-Movie Data:\n" +
                "\tMovie: " + list.get(k).getMovieURI() + "\n" +
                "\tActor: " + list.get(k).getActorURI() + "\n");
        }
    }
}
```

## bridges.data\_src\_dependent.ActorMovieWikidata Class Reference

### Detailed Description

Represent Actor Movie relations extacted from Wikidata.

Note that the end user will not create an object of that type in regular circumstances. But rather, `ActorMovieWikidata` objects are returned by `bridges::connect::DataSource::getWikidataActorMovie()`

### See also

Exemple of how to access that type of data is provided at: [https://bridgesuncc.github.io/tutorials/Data\\_WikiDataActor.html](https://bridgesuncc.github.io/tutorials/Data_WikiDataActor.html)

### Author

Erik Saule

### Public Member Functions

`ActorMovieWikidata (String movieURI, String actorURI, String movieName, String actorName)`

`String getMovieURI ()`  
get movie URI More...

`void setMovieURI (String movieURI)`  
set movie URI More...

`String getActorURI ()`  
get actor URI More...

`void setActorURI (String actorURI)`  
set actor URI More...

`String getMovieName ()`  
get movie name More...

`void setMovieName (String movieName)`  
set movie name More...

`String getActorName ()`  
get actor name More...

`void setActorName (String actorName)`  
set actor name More...

### Package Attributes

`String actorURI`

`String movieName`

`String actorName`

## Accessing Open Street Map Data: An Example BRIDGES program

[Java](#)

[C++](#)

[Python](#)

```
import bridges.connect.Bridges;
import bridges.connect.DataSource;
import bridges.data_src_dependent.OsmData;
import bridges.data_src_dependent.OsmVertex;
import bridges.data_src_dependent.OsmEdge;

// An example program fragment to illustrate how to retrieve Open Street map data.
// The program retrieves the map of UNC Charlotte campus, prints the number of vertices
// and edges, and the location of the first vertex

public class osm_snippet {
    public static void main(String[] args) throws Exception {

        //create the Bridges object
        Bridges bridges = new Bridges(YOUR_ASSIGNMENT_NUMBER, "YOUR_USER_ID", "YOUR_APT_KEY");

        DataSource ds = bridges.getDataSource();
        // OsmData osm_data = ds.getOsmData("Charlotte, North Carolina", "default");
        OsmData osm_data = ds.getOsmData(41.8313177632377, -80.2593749997456,
            42.008577297430456, -97.94531249997696);

        OsmVertex[] vertices = osm_data.getVertices();
        OsmEdge[] edges = osm_data.getEdges();

        System.out.println("Number of Vertices [Charlotte]: " + vertices.length);
        System.out.println("Number of Edges [Charlotte]: " + edges.length);

        // get cartesian coordinate location of first vertex
        double[] coords = osm_data.getVertices()[0].getCartesian_coord();
        System.out.println("Location of first vertex [Cartesian Coord]: " + coords[0] + " " +
            + coords[1]);
    }
}
```

## bridges.data\_src\_dependent.OsmData Class Reference

### Detailed Description

Class that hold Open Street Map vertices.

This class holds Open Street Map data, from <https://openstreetmap.org>

Objects from this class are typically not created by the user but retruned by `bridges::connect::DataSource::getOsmData()`

Check out how to use OSM data at: [https://bridgesuncc.github.io/tutorials/Data\\_OSM.html](https://bridgesuncc.github.io/tutorials/Data_OSM.html)

#### Author

Kalpathi Subramanian, Erik Saulé

#### Date

2/16/19, 12/26/20

### Public Member Functions

<code>OsmData()</code>	<code>OsmData(OsmVertex[] vertices, OsmEdge[] edges, String name)</code>
<code>OsmVertex[] getVertices()</code>	<code>void setVertices(OsmVertex[] vertices)</code>
<code>String getName()</code>	<code>void setName(String name)</code>
<code>OsmEdge[] getEdges()</code>	<code>void setEdges(OsmEdge[] edges)</code>
<code>double[] getCartesianRangeY()</code>	<code>double[] getLatitudeRange()</code>
<code>double[] getLongitudeRange()</code>	<code>double[] getCartesianRangeX()</code>
<code>void getLatLongRange(double[] latr, double[] lonr)</code>	<code>GraphAdjList&lt; Integer, OsmVertex, Double &gt; getGraph()</code>

### Package Attributes

<code>double[] longitude_range</code>
<code>double[] cartesian_range_x</code>
<code>double[] cartesian_range_y</code>

[bridgesuncc.github.io/tutorials/Data\\_Amenity.html](https://bridgesuncc.github.io/tutorials/Data_Amenity.html)

### Step 1: Suggested Examples:

```
data_source.get_amenity_data("New York, New York", "food")
data_source.get_amenity_data("Los Angeles, California", "school")
data_source.get_amenity_data("Chicago, Illinois", "firestation")
```

The program snippet below illustrates how to obtain map data of the UNC Charlotte.

## Accessing Open Street Map Amenity Data: An Example BRIDGES program

Java

C++

Python

```
import java.lang.String;
import bridges.base.SElement;
import bridges.base.GraphAdjList;
import bridges.connect.Bridges;
import bridges.connect.DataSource;
import bridges.data_src_dependent.Amenities;
import bridges.data_src_dependent.AmenityData;

public class osm_amenity_snippet {
    public static void main(String[] args) throws Exception {

        // initialize Bridges
        Bridges bridges = new Bridges(YOUR_ASSIGNMENT_NUMBER, "YOUR_USER_ID",
            "YOUR_API_KEY");

        // title, description
        bridges.setTitle("An Open Street Map Example");

        // get the OSM data
        DataSource ds = bridges.getDataSource();
        AmenityData amenity_data = ds.getAmenityData(38.77657, -77.20918, 39.03199, -76.8999, "food");

        System.out.println("Total Points: " + amenity_data.getCount());
        System.out.println("ID: " + amenity_data.getAmenities().get(0).getId());
        System.out.println("Name: " + amenity_data.getAmenities().get(0).getName());
        System.out.println("Lat: " + amenity_data.getAmenities().get(0).getLat());
        System.out.println("Lon: " + amenity_data.getAmenities().get(0).getLon());

    }
}
```

[https://bridgesuncc.github.io/doc/java-api/current/html/bridges/data\\_src\\_dependent/Amenities.html](https://bridgesuncc.github.io/doc/java-api/current/html/bridges/data_src_dependent/Amenities.html)

Public Member Functions | List of all members

## bridges.data\_src\_dependent.Amenities Class Reference

### Detailed Description

Class that hold individual Open Street Map [Amenities](#).

This class holds the individual information for each amenity requested.

Usually this class is not created by the user but part of an [AmenityData](#) object returned by `bridges::connect::DataSource::getAmenityData()`

Check out the tutorial on getting amenity data at [https://bridgesuncc.github.io/tutorials/Data\\_Amenity.html](https://bridgesuncc.github.io/tutorials/Data_Amenity.html)

### Author

Jay Strahler

### Public Member Functions

<code>Amenities()</code>	<code>Amenities(double id_val, double lat, double lon, String name, String[] other)</code>
<code>double getId()</code>	<code>void setId(double id_val)</code>
<code>double getLat()</code>	<code>void setLat(double lat)</code>
<code>double getLon()</code>	<code>void setLon(double lon)</code>
<code>String getName()</code>	<code>void setName(String name)</code>
<code>String[] getOther()</code>	<code>void setOther(String[] other)</code>

### Constructor & Destructor Documentation

◆ `Amenities()` [1/2]

`bridges.data_src_dependent.Amenities.Amenities( )`

Default Constructor

Reddit

The slide will come, one day! :)

# Table of Contents

1 Datasets

2 Visualizations

3 Assignments

# Arrays

Bridges-Java: bridges.data\_src\_dependent.Song Class Reference — Mozilla Firefox

Bridges - Array Tutorial — Mozilla Firefox

[bridgesuncc.github.io/tutorials/Array.html](https://bridgesuncc.github.io/tutorials/Array.html)

```
import bridges.base.Array1D;

public class array1d {
    public static void main(String[] args) throws Exception {

        //create the Bridges object, set credentials
        Bridges bridges = new Bridges(YOUR_ASSIGNMENT_NUMBER, "YOUR_USER_ID",
            "YOUR_API_KEY");

        // title, description
        bridges.setTitle("One-Dimensional Array Example");
        bridges.setDescription("One dimensional array with visual attributes");

        // set array dimensions, allocate array of elements
        int arraySize = 10;

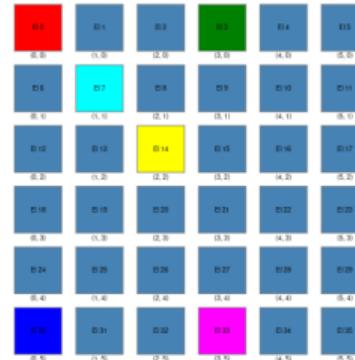
        Array1D<Integer> arr = new Array1D<Integer>(arraySize);

        // populate the array, with squares of indices
        // use the values to label the elements
        for (int k = 0; k < arr.getSize(); k++) {
            arr.getElement(k).setValue(k * k);
            arr.getElement(k).setLabel(String.valueOf(arr.getElement(k).getValue()));
        }

        // color the array elements
        arr.getElement(0).setColor("red");
        arr.getElement(1).setColor("green");
        arr.getElement(2).setColor("blue");
        arr.getElement(3).setColor("cyan");
        arr.getElement(4).setColor("magenta");
        arr.getElement(5).setColor("yellow");
        arr.getElement(6).setColor("red");
        arr.getElement(7).setColor("green");
        arr.getElement(8).setColor("blue");
        arr.getElement(9).setColor("black");

        // tell Bridges what datastructure to visualize
        bridges.setDataSource(arr);

        // visualize the list
        bridges.visualize();
    }
}
```



(Slice: 0)



(Slice: 1)



(Slice: 2)

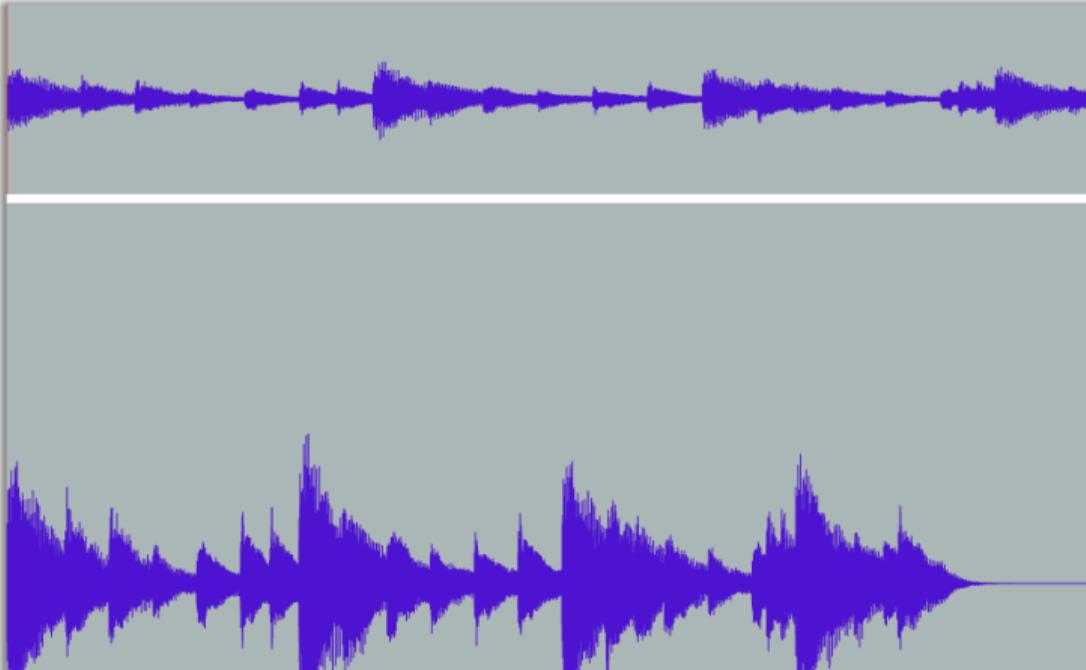


# AudioClip

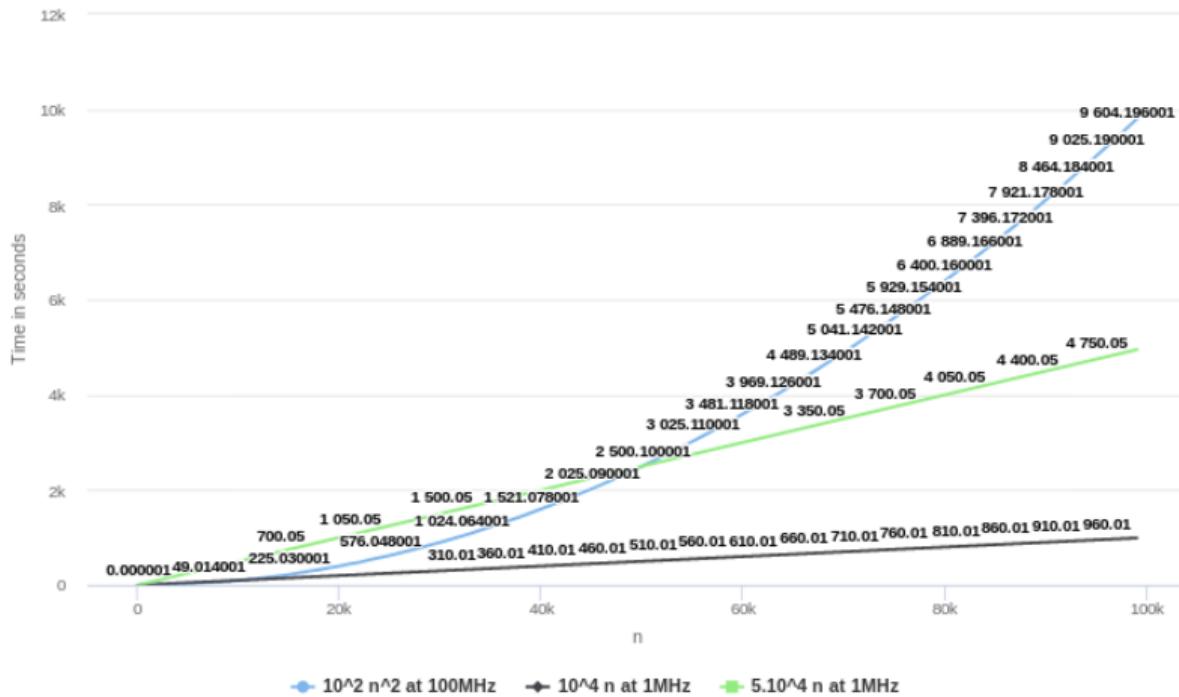
## Audio Mixing

Mixing two AudioClip's together

33.00



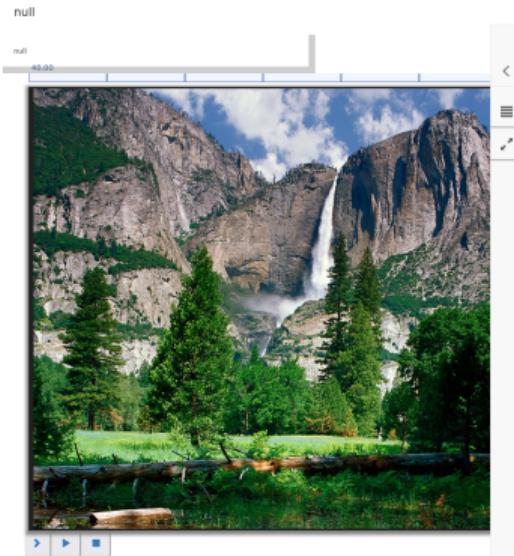
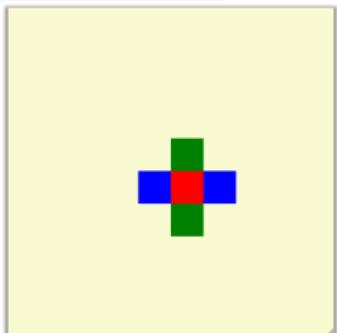
# LineChart



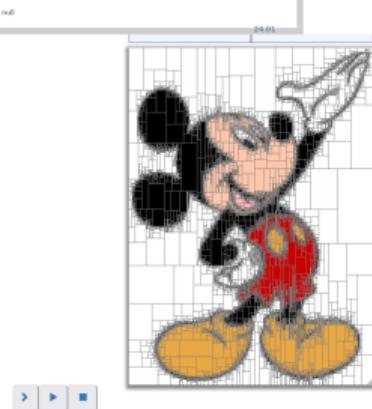
# ColorGrid

BRIDGES Color Grid Tutorial - Part 1

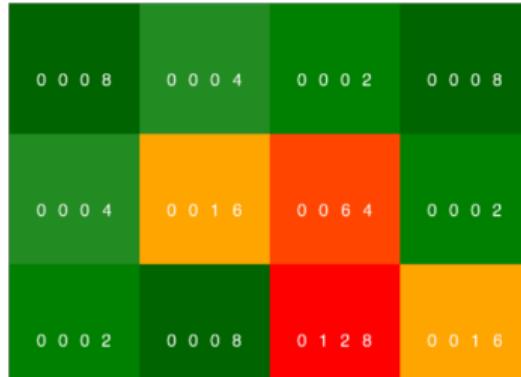
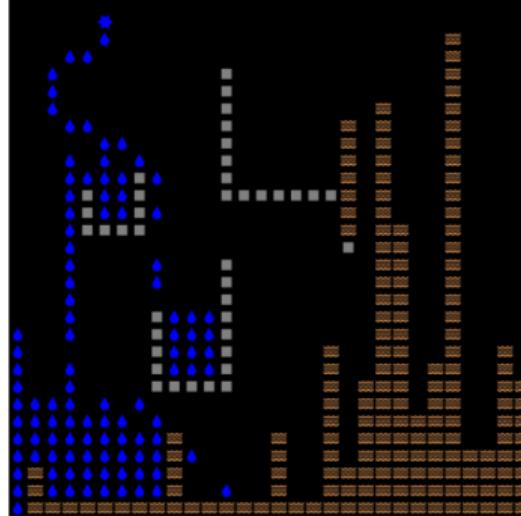
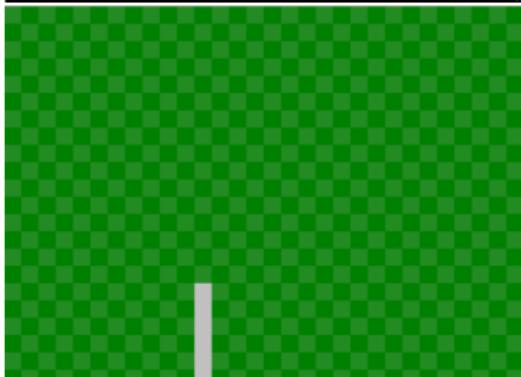
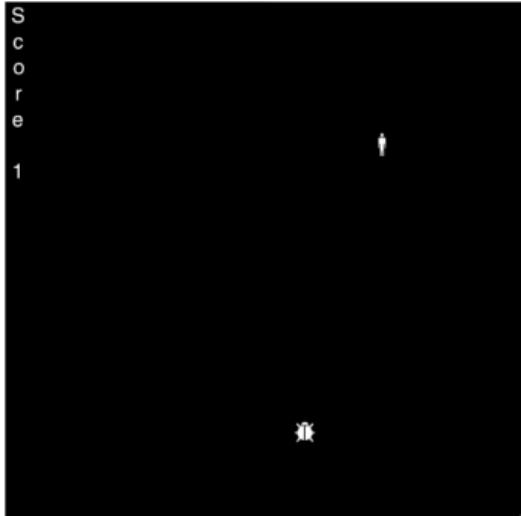
This is an example of the color grid with height and width of 10 units each.



KD Tree Representation of an Image



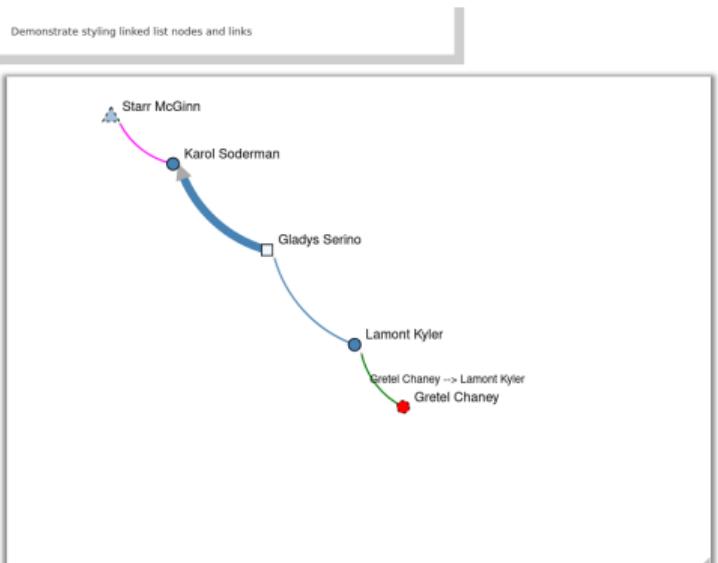
# Games



# Linked List

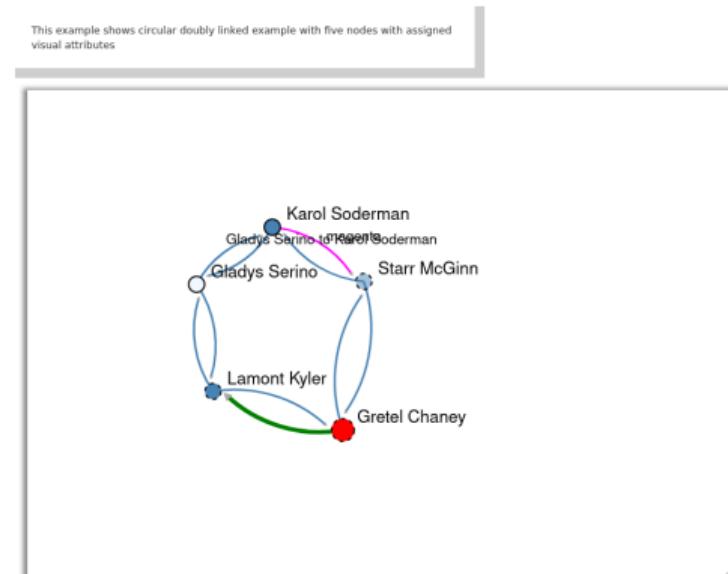
## A Singly Linked List Example

Demonstrate styling linked list nodes and links



## A Circular Doubly Linked List Example

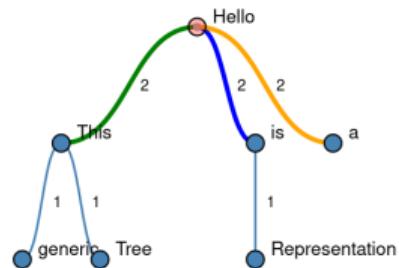
This example shows circular doubly linked example with five nodes with assigned visual attributes



# Trees

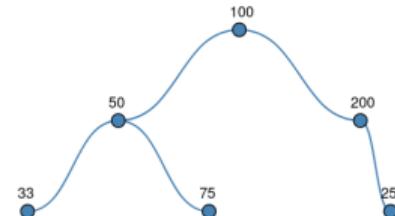
## A General Tree Example

A basic tree with seven nodes. Three on one side and three on the other. The root node is set to red with 0.3 opacity. The other six nodes are neutral color.



## A Simple Binary Search Tree Example

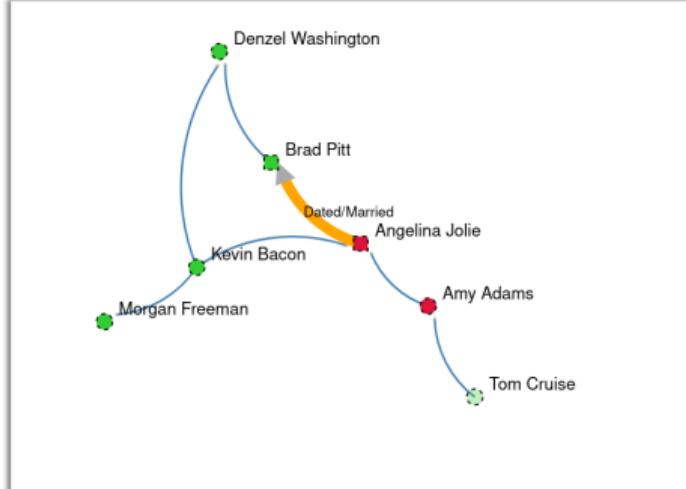
This example illustrates a binary search tree built using BRIDGES



# Graphs

A Simple Adjacency list based Graph Example.

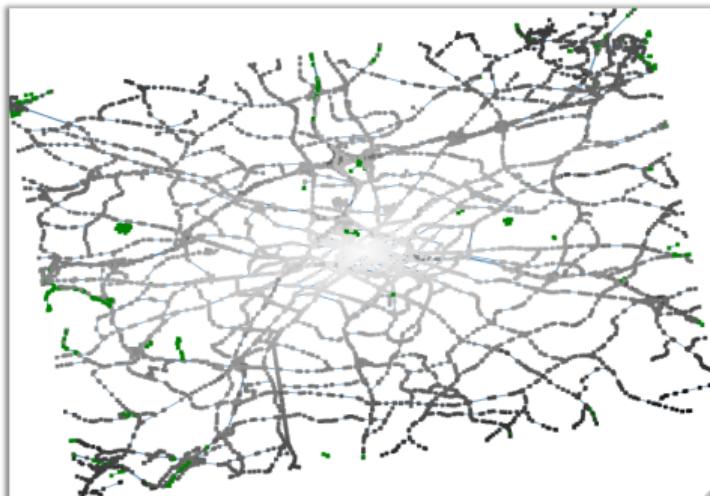
Demonstrate styling graph nodes and links with visual attributes



Graph : OpenStreet Map Example

Charlotte: Color of vertices is based on distance from center of the city

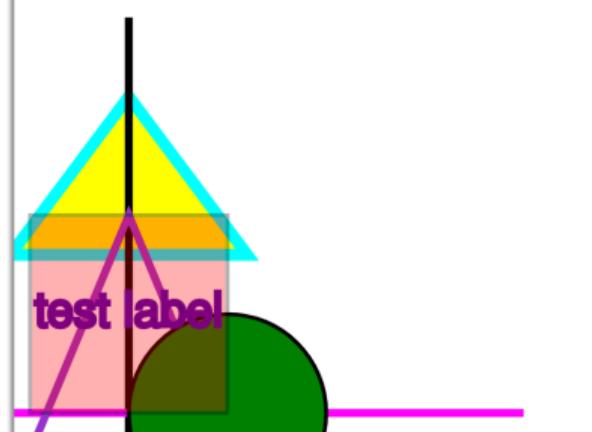
9.00



# Shape Collection

## Symbol Collection

Red square, green circle, magenta horizontal and vertical lines, and a test label with a purple outline.



Proc 2

C

F

Proc 1

B

E

H

G

J

Proc 0

A

D

I

K

L

time

## PQBook

MinHeap represented as a Binary Tree.

8.00



# Table of Contents

1 Datasets

2 Visualizations

3 Assignments

# Assignment Repository

Structured by course and topics: **[BRIDGES Assignments]**