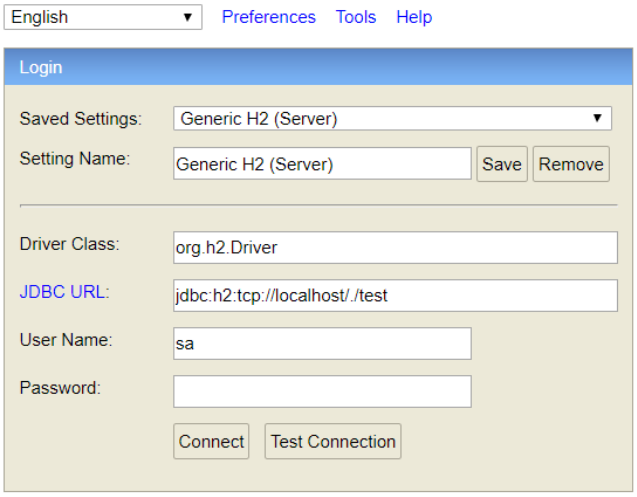
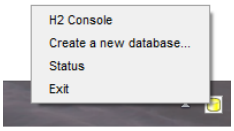
In this project the basic use of Hibernate will be presented

1. Setup a local database.
    - Visit the page of H2 database Engine and download the "All Platforms" version
    - Unpack the zip file to a preferred location
    - Double clicking on the jar file found at ./h2/bin starts the local database
    - A page like below should open in a new browser tab



2. Open the starting Maven Project
    - Download the project from the page of the class
    - In Netbeans select File -> Import Project… -> From Zip
3. Examine `pom.xml` file
    - In pom.xml you should see target and source versions matching to your installed Java platform
    - Note the `com.h2database.h2` and `org.hibernate.hibernate-core` dependencies
4. Setup the DB configuration
    - Examine the `hibernate.cfg.xml` file. Here you can setup the way how you reach you database.
        – If you run H2 in Embedded mode be careful, to close the running .jar file, before you run your application.
        – If you run the server mode you have to create a database first. Right click on the tray icon of H2 and select Create new Database.



– Create a database with a preferred name and copy the generated JDBC url to the clipboard.

– Copy the url to the hibernate.cfg.xml file to the line

<property name="connection.url">

– Also set the username and password you have given

- Discuss the meaning of the remaining lines of the file
- Now if you run the project it should work

5. Study how in the main app Student data is managed in sessions.
6. Create your own ORM mappings following the below description
7. Create a new Class under the `hibernate.entity` package. The name of it is Animal and it has the following properties:
   - gender (`enum` MALE/FEMALE),
   - age (`int`),
   - name (`String`),
   - id (`int`)

   Add getters and setters to these fields. Also add a default constructor.
8. Use persistence annotations to create the animal entity
   - Use `@Entity` and `@Table(name = "animal")` on the `Animal` class
   - Use `@Id,` and `@GeneratedValue(strategy = GenerationType.AUTO)` on the `id` field
   - Use `@Column(name = "…", unique = … , nullable = false)` on all fields. Set unique to true for the id, and for false for the other fields.
9. In the hibernate.cfg.xml file under Mappings add the Animal class. The following line is added in the source: <mapping class="net.javaguides.hibernate.entity.Animal" />
10. In the main method create a new hibernate session using the `HibernateUtil` class
11. Instantiate a new `Animal` and save it to the database
    - Instantiate and initialize the animal object.
    - Save the animal using the code below:

```
session.beginTransaction();
session.save(elephant);
session.getTransaction().commit();
session.close();
HibernateUtil.closeSessionFactory();
```

12. Check the new data appearing in the database

Advanced project:

1. Create a DAO (Data Access Object) class for the Animal class
   - Create a new class under the `hibernate.db` package name: `AnimalDAO`
   - Make the class implement the `AutoCloseable` interface
   - Open a new Session in the constructor, and close it in the `close()` method
   - Add simple transaction methods to add, delete and update `Animal` objects
   - Add a method to get all the saved animals. Use the code below:

```
public List<Animal> getAnimals(){
    String hql = "FROM hibernatetest.model.Animal";
    Query query = session.createQuery(hql);
    return query.list();
}
```

   Read this for more about Hibernate Query Language:
   http://www.tutorialspoint.com/hibernate/hibernate_query_language.htm
2. Modify the main method so that it uses the new DAO class for DB operations

Advanced project 2 - Mapping collections:

1. Add a `Zoo` class to your project. A zoo has the following fields: `id`, `name`, `animals`. This latter field is a `Set` of animals held by the zoo.
2. Annotate the `Zoo` class to be the zoo entity. Use the following annotations on the animals

```
@OneToMany(cascade = CascadeType.ALL)
@JoinColumn(name = "zoo_id")
private Set<Animal> animals;
```

   field:
3. In the main method instantiate a new `Zoo` class and add an animal to it. Save the `Zoo`. Note that the `Animal` table also has been updated.

   ---
4. Modify the `Zoo` class by removing the animal field and the depending methods from it.
5. Add a new field to the `Animal` class. Name it `owner_zoo`. The type is `Zoo`.

1. Use these annotations on the field:

```
@ManyToOne
@Cascade(CascadeType.ALL)
@JoinColumn(name = "zoo_id")
private Zoo owner_zoo;
```

2. Create an animal and a zoo. Set the `owner_zoo` field of the animal. Persist the animal.

See more about Hibernate associations:

http://viralpatel.net/blogs/hibernate-one-to-many-annotation-tutorial/