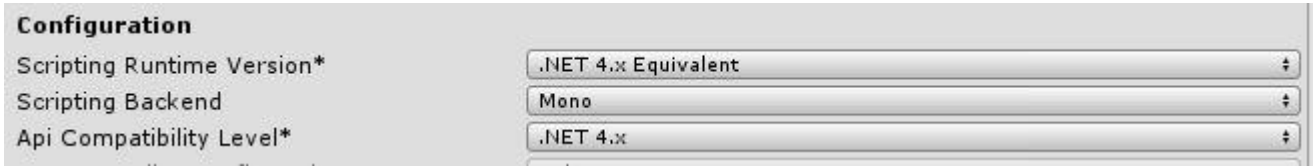
Tables Data Utility

Simple structured data storage.
Store player progress, create databases…

Contact: uutilsplugin@gmail.com

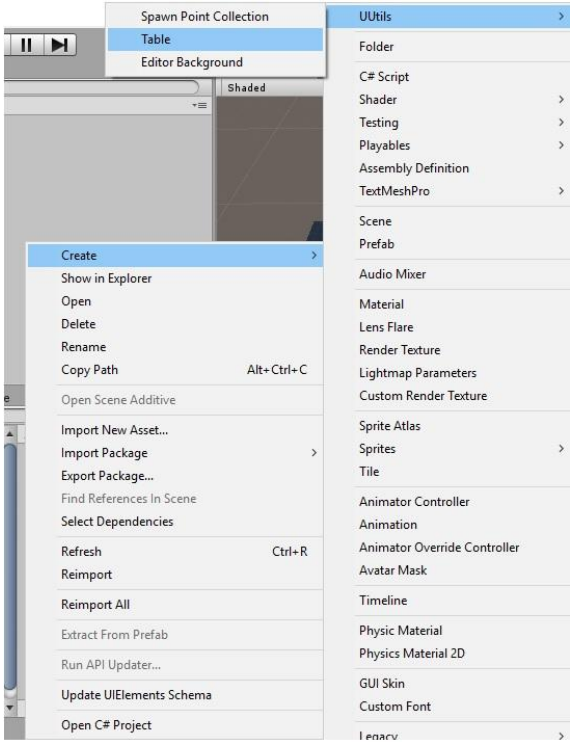
.NET version

- To be able to use this plugin, you need to change you project configuration
- Go to Edit->Project Settings->Player and scroll down to Configuration
- Change to the settings in the picture
- This is required as this plugin uses some features which are not included in lower versions



Create table

- To create a table scriptable object, right click anywhere in your project folder, Create -> UUtils -> Table
- Then, open “Table Editor” located in top bar under UUtils -> Table Editor

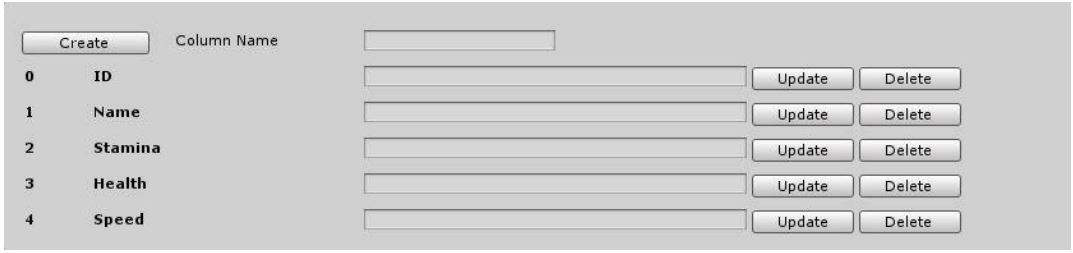


Editor

- Load a previously created table by placing it in the “Table SO” field and clicking “Load”
- Top right corner contains a console which displays messages on actions which happened during editing
- Top left bar contains three buttons:
  - “Convert” - converts .json table
  - “Browse” - current menu, displays all data in a table
  - “Structure” - add columns to a table

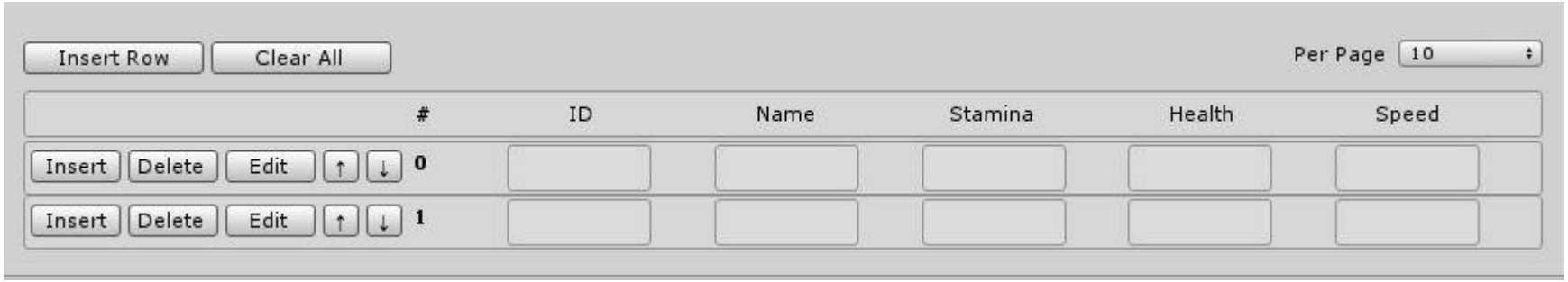
Structure

- Click on the “Structure” button to open the menu
- To create a column, write its name and click “Create”
- Column will be created and added below, you can see a log of it in the top right corner
- To delete a column, click the “Delete” next to it
- To update its name, write a new name and click “Update”
- Objects can subscribe to the Update so they can be notified when a column name has changed, that will be covered later in code



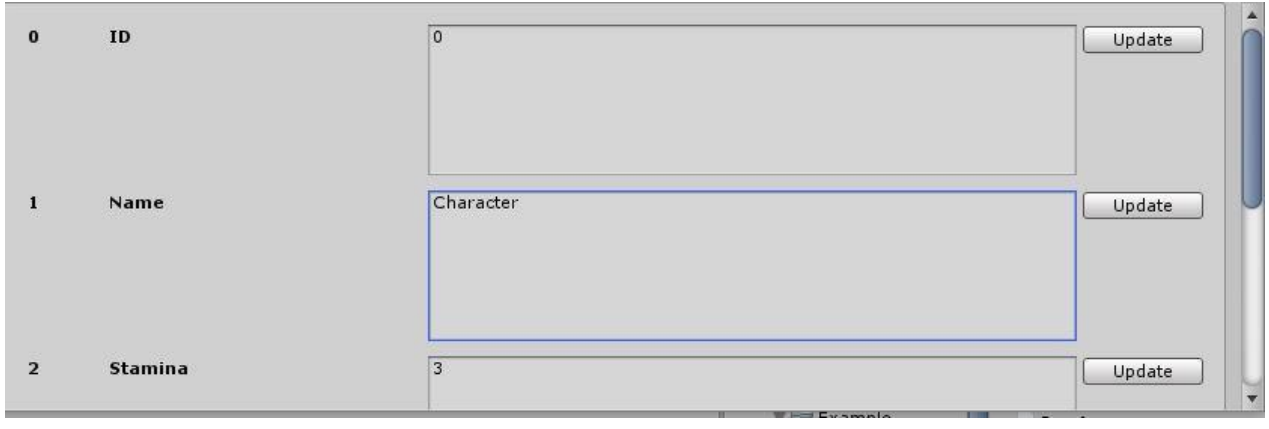
Browse

- Click on the “Browse” button to open the menu
- To insert a new row, click “Insert Row”
- “Clear All” deletes all items (warning is displayed before)
- Every new row has options
  - “Insert” - inserts a new row before that one
  - “Delete” - deletes the row
  - “Edit” - opens edit menu
  - Up and down arrows shift row position
- Click “Edit” on any row you want to edit



Edit Row

- To update a value in the rows column, simply write it, value is automatically updated
- You can click the “Update” button to notify any subscribed objects of the change, this will be covered later

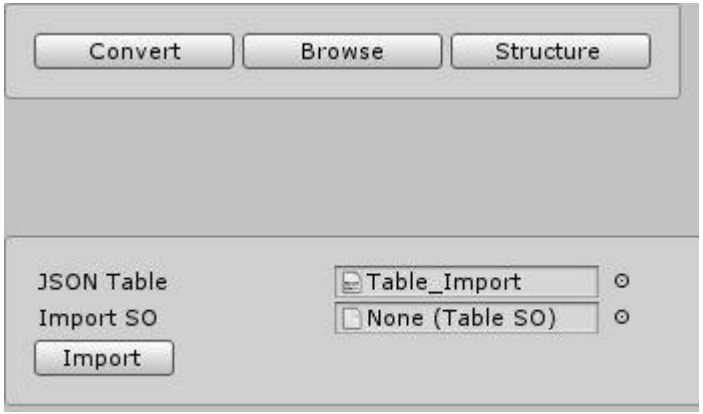


Save .JSON

- Back in the “Browse” menu, click “Save” if you’d like to export the Table as a “.json” file
- Click “Save JSON And Reload” to save the file and reload unity files, this will import the file if you saved it somewhere in this project, you can also use the “Save” option and just press Ctrl+R
- “TableSO” field can be empty as you’re saving the currently loaded table
- Unity’s JsonUtility is used to serialize the table

Convert .JSON

- Open “Convert” menu
- Here you can import a previously created “.json” table
- Select the “.json” file you want to import and create a new empty “TableSO” by right clicking somewhere in the project, then Create->UUtils->Table and place that “TableSO” as the “Import SO”
- Click “Import”



Code & Files: Very Short Explanation

- TableSO.cs
  - every time you create a new “TableSO” from the UI, you’re creating a scriptable object
  - This file contains a “Table” field for “Table” class
- Table.cs
  - contains all data
  - Data is stored in columns
  - each column has a row and all columns have an identical row count
  - This way you can easily get all rows from a single column just by getting the column by its name
  - When you want to get an entire row by index, each row in a column at that index will be combined into a class TableRow.cs
  - As this plugins main purpose is simplicity, values are currently only stored as strings
  - There’s a possibility this will get upgraded

Code & Files: Very Short Explanation

- ITableColumn.cs
  - Interface for performing operations on a single column with all row values for the column
- ColumnArgs.cs
  - Event args used when ITableColumn is updated
- ITableRowValue.cs
  - Interface for performing operations on a single column in a row
- RowArgs.cs
  - Event args used when ITableRowValue is updated
- TableRow.cs
  - Class for getting an entire row

These were classes and interfaces that you should use.

API documentation

- API documentation was created with Doxygen

Example folder

- Contains TableSO assets used in the manual
- Contains a ExampleTables.cs which shows you codes you should use and how to use them
- There’s no test scene