GWT - CELLTABLE WIDGET

Introduction

The **CellTable** widget represents a A tabular view that supports paging and columns.

Class declaration

Following is the declaration for **com.google.gwt.user.cellview.client.CellTable<T>** class:

```
public class CellTable<T>
   extends AbstractHasData<T>
```

Class constructors

S.N.	Constructor & Description
1	**CellTable** Constructs a table with a default page size of 15.
2	**CellTable***intpageSize* Constructs a table with the given page size.
3	**CellTable***intpageSize, CellTable. Resourcesresources* Constructs a table with the given page size with the specified CellTable.BasicResources.
4	**CellTable***intpageSize, CellTable. Resourcesresources, ProvidesKey < T > keyProvider* Constructs a table with the given page size, the specified CellTable.BasicResources, and the given key provider.
5	**CellTable***intpageSize, ProvidesKey < T > keyProvider* Constructs a table with the given page size and the given key provider.
6	**CellTable***ProvidesKey < T > keyProvider* Constructs a table with a default page size of 15, and the given key provider.

Class methods

S.N.	Function name & Description
1	**void addColumn***Column < T, ? > col*

Adds a column to the table.

2

void addColumn*Column < T, ? > col, Header < ? > header*

Adds a column to the table with an associated header.

3

void addColumn*Column < T, ? > col, Header < ? > header, Header < ? > footer*

Adds a column to the table with an associated header and footer.

4

void addColumn*Column < T, ? > col, SafeHtmlheaderHtml*

Adds a column to the table with an associated SafeHtml header.

5

void addColumn*Column < T, ? > col, SafeHtmlheaderHtml, SafeHtmlfooterHtml*

Adds a column to the table with an associated SafeHtml header and footer.

6

void addColumn*Column < T, ? > col, java. lang. StringheaderString*

Adds a column to the table with an associated String header.

7

void add Column *Column < T, ? > col, java. lang. StringheaderString, java. lang. StringfooterString*

Adds a column to the table with an associated String header and footer.

8

void addColumnStyleName*intindex, java. lang. StringstyleName*

Add a style name to the TableColElement at the specified index, creating it if necessary.

9

protected Element convertToElements*SafeHtmlhtml*

Convert the specified HTML into DOM elements and return the parent of the DOM elements.

10

protected boolean dependsOnSelection

Check whether or not the cells in the view depend on the selection state.

11

protected void do Selection *Eventevent, Tvalue, introw, intcolumn*

Deprecated. use Abstract Has Data.add Cell Preview Handler *com. google. gwt. view. client. CellPreviewEvent. Handler* instead.

12

int getBodyHeight

Return the height of the table body.

13

protected Element getChildContainer

Return the element that holds the rendered cells.

14

int getHeaderHeight

Return the height of the table header.

15

protected Element getKeyboardSelectedElement

Get the element that has keyboard selection.

16

TableRowElement getRowElement*introw*

Get the TableRowElement for the specified row.

17

protected boolean isKeyboardNavigationSuppressed

Check if keyboard navigation is being suppressed, such as when the user is editing a cell.

18

protected void onBlur

Called when the widget is blurred.

19

protected void onBrowserEvent2*Eventevent*

Called after AbstractHasData.onBrowserEvent*Event* completes.

20

protected void onFocus

Called when the widget is focused.

21

void redrawFooters

Redraw the table's footers.

22

void redrawHeaders

Redraw the table's headers.

23

void removeColumn*Column* < *T*, ? > *col*

Remove a column.

24

void removeColumn*intindex*

Remove a column.

25

void removeColumnStyleName*intindex, java. lang. StringstyleName*

Remove a style from the TableColElement at the specified index.

26

protected void renderRowValues
SafeHtmlBuildersb, java. util. List < T > values, intstart, SelectionModel < ?superT > selectionModel

Render all row values into the specified SafeHtmlBuilder.

27

protected void replaceAllChildren*java. util. List < T > values, SafeHtmlhtml*

Replace all children with the specified html.

28

protected boolean resetFocusOnCell

Reset focus on the currently focused cell.

29

protected void setKeyboardSelected*intindex, booleanselected, booleanstealFocus*

Update an element to reflect its keyboard selected state.

30

void setRowStyles*RowStyles < T > rowStyles*

Sets the object used to determine how a row is styled; the change will take effect the next time that the table is rendered.

31

protected void setSelected*Elementelem, booleanselected*

Deprecated. this method is never called by AbstractHasData, render the selected styles in renderRowValues*SafeHtmlBuilder*, *List*, *int*, *SelectionModel*

Methods inherited

This class inherits methods from the following classes:

- com.google.gwt.user.client.ui.UIObject
- com.google.gwt.user.client.ui.Widget
- com.google.gwt.user.cellview.client.AbstractHasData
- java.lang.Object

CellTable Widget Example

This example will take you through simple steps to show usage of a CellTable Widget in GWT. Follow the following steps to update the GWT application we created in *GWT - Create Application* chapter:

Step	Description
1	Create a project with a name *HelloWorld* under a package *com.tutorialspoint* as

explained in the *GWT - Create Application* chapter.

2 Modify *HelloWorld.gwt.xml*, *HelloWorld.css*, *HelloWorld.html* and *HelloWorld.java* as explained below. Keep rest of the files unchanged.

3 Compile and run the application to verify the result of the implemented logic.

Following is the content of the modified module descriptor **src/com.tutorialspoint/HelloWorld.gwt.xml**.

```
<?xml version="1.0" encoding="UTF-8"?>
<module rename-to='helloworld'>
  <!-- Inherit the core Web Toolkit stuff.                        -->
  <inherits name='com.google.gwt.user.User'/>

  <!-- Inherit the default GWT style sheet.                       -->
  <inherits name='com.google.gwt.user.theme.clean.Clean'/>

  <!-- Specify the app entry point class.                         -->
  <entry-point class='com.tutorialspoint.client.HelloWorld'/>

  <!-- Specify the paths for translatable code                    -->
  <source path='client'/>
  <source path='shared'/>

</module>
```

Following is the content of the modified Style Sheet file **war/HelloWorld.css**.

```
body{
   text-align: center;
   font-family: verdana, sans-serif;
}
h1{
   font-size: 2em;
   font-weight: bold;
   color: #777777;
   margin: 40px 0px 70px;
   text-align: center;
}
```

Following is the content of the modified HTML host file **war/HelloWorld.html**.

```
<html>
<head>
<title>Hello World</title>
   <link rel="stylesheet" href="HelloWorld.css"/>
   <script language="javascript" src="helloworld/helloworld.nocache.js">
   </script>
</head>
<body>

<h1>CellTable Widget Demonstration</h1>
<div ></div>

</body>
</html>
```

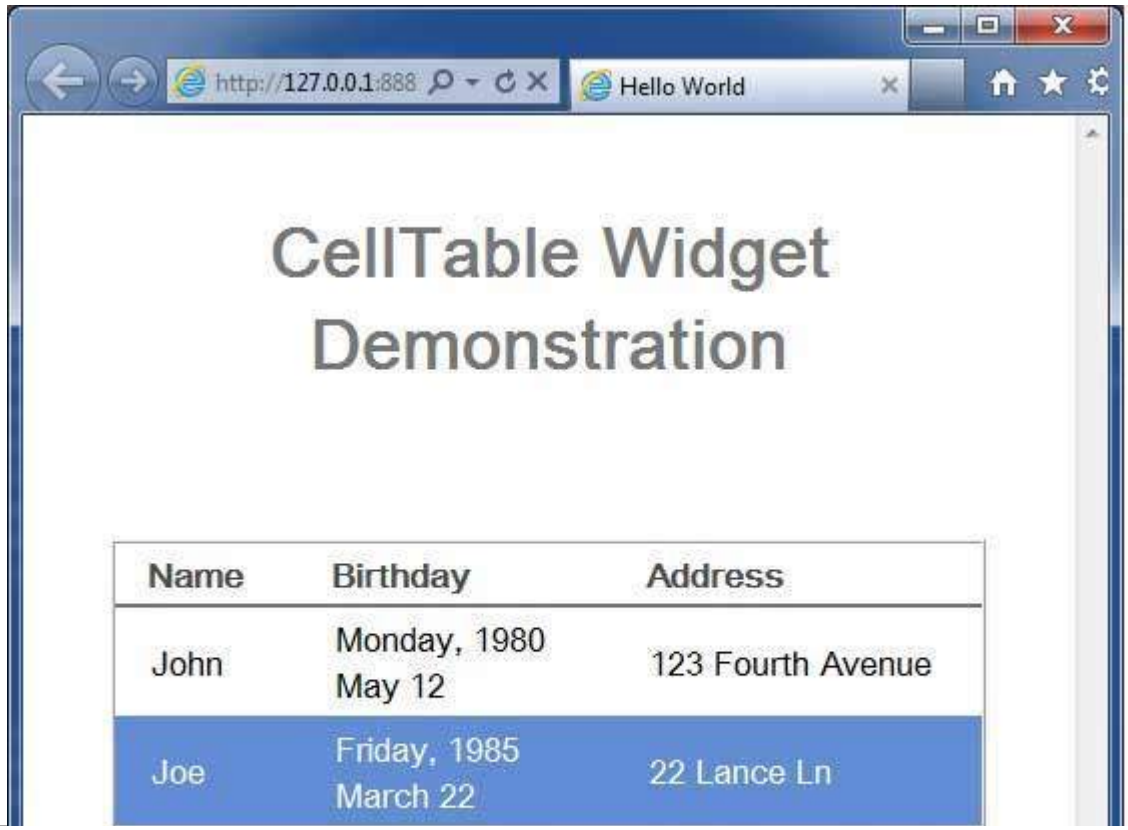
Let us have following content of Java file **src/com.tutorialspoint/HelloWorld.java** which will demonstrate use of CellTable widget.

```
package com.tutorialspoint.client;

import java.util.Arrays;
import java.util.Date;
```

```
import java.util.List;

import com.google.gwt.cell.client.DateCell;
import com.google.gwt.core.client.EntryPoint;
import com.google.gwt.user.cellview.client.CellTable;
import com.google.gwt.user.cellview.client.Column;
import com.google.gwt.user.cellview.client
.HasKeyboardSelectionPolicy.KeyboardSelectionPolicy;
import com.google.gwt.user.cellview.client.TextColumn;
import com.google.gwt.user.client.Window;
import com.google.gwt.user.client.ui.RootPanel;
import com.google.gwt.user.client.ui.VerticalPanel;
import com.google.gwt.view.client.SelectionChangeEvent;
import com.google.gwt.view.client.SingleSelectionModel;

public class HelloWorld implements EntryPoint {
   /**
   * A simple data type that represents a contact.
   */
   private static class Contact {
      private final String address;
      private final Date birthday;
      private final String name;

      public Contact(String name, Date birthday, String address) {
         this.name = name;
         this.birthday = birthday;
         this.address = address;
      }
   }

   /**
   * The list of data to display.
   */
   private static final List<Contact> CONTACTS = Arrays.asList(
      new Contact("John", new Date(80, 4, 12), "123 Fourth Avenue"),
      new Contact("Joe", new Date(85, 2, 22), "22 Lance Ln"),
      new Contact("George",new Date(46, 6, 6),"1600 Pennsylvania Avenue"));

   public void onModuleLoad() {
      // Create a CellTable.
      CellTable<Contact> table = new CellTable<Contact>();
      table.setKeyboardSelectionPolicy(KeyboardSelectionPolicy.ENABLED);

      // Add a text column to show the name.
      TextColumn<Contact> nameColumn =
      new TextColumn<Contact>() {
         @Override
         public String getValue(Contact object) {
            return object.name;
         }
      };
      table.addColumn(nameColumn, "Name");

      // Add a date column to show the birthday.
      DateCell dateCell = new DateCell();
      Column<Contact, Date> dateColumn
      = new Column<Contact, Date>(dateCell) {
         @Override
         public Date getValue(Contact object) {
            return object.birthday;
         }
      };
      table.addColumn(dateColumn, "Birthday");

      // Add a text column to show the address.
      TextColumn<Contact> addressColumn
      = new TextColumn<Contact>() {
         @Override
```

```
            public String getValue(Contact object) {
               return object.address;
            }
         };
         table.addColumn(addressColumn, "Address");

         // Add a selection model to handle user selection.
         final SingleSelectionModel<Contact> selectionModel
         = new SingleSelectionModel<Contact>();
         table.setSelectionModel(selectionModel);
         selectionModel.addSelectionChangeHandler(
         new SelectionChangeEvent.Handler() {
            public void onSelectionChange(SelectionChangeEvent event) {
               Contact selected = selectionModel.getSelectedObject();
               if (selected != null) {
                  Window.alert("You selected: " + selected.name);
               }
            }
         });

         // Set the total row count. This isn't strictly necessary,
         // but it affects paging calculations, so its good habit to
         // keep the row count up to date.
         table.setRowCount(CONTACTS.size(), true);

         // Push the data into the widget.
         table.setRowData(0, CONTACTS);

         VerticalPanel panel = new VerticalPanel();
         panel.setBorderWidth(1);
         panel.setWidth("400");
         panel.add(table);

         // Add the widgets to the root panel.
         RootPanel.get().add(panel);
      }
   }
```

Once you are ready with all the changes done, let us compile and run the application in development mode as we did in GWT - Create Application chapter. If everything is fine with your application, this will produce following result:



George	Saturday, 1946 July 06	1600 Pennsylvania Avenue