SessionParams

# Introduction

The SessionParams class is Server-side code. It keeps track of variables required for sorting, paging, filtering and other information if required. The idea is to persist this kind of data across web pages for each web user.

It uses a viewstate method. And places the encoded data on the webpage. This is so we are not using server memory for large websites.

# How it works

The must be passed to the SessionParams class using the parameter Info class, which was passed from the webpage.

Therefore, to get the SessionParams out of cache, we must have the “rocketsessionkey” rendered on the html page and placed into a simplisity paramField.

Example:

// Pass SessionParams back to the server

simplisity\_setParamField('simplisitysessionparams', '@(Model.SessionParamViewState)');

*This could have been done with a cookie, but cross site calls can be awkward with cookies, so we have decided to make the method a field on the html page and reply with simplisity params.*

In the above example we place the “RocketSessionKey” into a “rocketsessionkey” field. All param fields are passed back to the server from the client and used to populate the SimplisityInfo class paramInfo. We can then use this class to populate the SessionParams.

### What happens when SessionParams is initiated

On initialization of the SessionParams we pass the paramInfo which came from the client on the API.

Header = new SessionParams(paramInfo);

The paramInfo will contain the “RocketSessionKey” that we rendered on the page, the SessionParams class will use that field (“genxml/hidden/rocketsessionkey”) to get the SessionParams from cache.

But this is only the first step, to be useful we must change the values storage and again save to memory.

The SessionParams class reads all the data fields in the paramInfo class, it uses this to overwrite the existing fields. Any field name/id can be used, but we have static defined ones, to deal with standard operations.

### Standard SessionParams Fields

string HeaderCacheKey = Info.GUIDKey

//Order by

string OrderBySQL = Info.GetXmlProperty("genxml/hidden/orderbysql")

int SortActivate = Info.GetXmlPropertyInt("genxml/hidden/sortorderactivate")

// Paging

int PageSize = Info.GetXmlPropertyInt("genxml/hidden/pagesize")

int Page = Info.GetXmlPropertyInt("genxml/hidden/page")

int RowCount = Info.GetXmlPropertyInt("genxml/hidden/rowcount")

//Filter SQL

string FilterSQLref = Info.GetXmlProperty("genxml/hidden/filtersqlref")

After the values in the header have been updated by the paramInfo, the SessionParams will save itself to cache.

# Filtering and Sorting Data Lists

The SessionParams is also used to keep track of filtering and sorting data. Each DNN Rocket module can have settings to define the filtering and sorting.