



SessionPartition Class

Contains methods to manage cache values in the session cache of a specific partition.

Namespace

[Cache](#)

Usage

This class extends [Cache.Partition](#) and inherits all of its non-static methods. Utility methods for creating and validating keys are not supported and can be called only from the `Cache.Partition` parent class. For a list of `Cache.Partition` methods, see [Partition Methods](#).

To get a session partition, call `Cache.Session.getPartition` and pass in a fully qualified partition name, as follows.

```
Cache.SessionPartition sessionPartition = Cache.Session.getPartition('namespace.myPartition')
```

See [Cache Key Format for Partition Methods](#).

Example

This class is the controller for a sample Visualforce page (shown in the subsequent code sample). The controller shows how to use the methods of `Cache.SessionPartition` to manage a cache value on a particular partition. The controller takes inputs from the Visualforce page for the partition name, key name for a counter, and initial counter value. The controller contains default values for these inputs. When you click **Rerender** on the Visualforce page, the `go()` method is invoked and increases the counter by one. When you click **Remove Key**, the counter key is removed from the cache. The counter value gets reset to its initial value when it's re-added to the cache.

```
public class SessionPartitionController {  
  
    // Name of a partition in the local namespace  
    String partitionInput = 'local.myPartition';  
    // Name of the key  
    String counterKeyInput = 'counter';  
    // Key initial value  
    Integer counterInitValue = 0;  
    // Session partition object  
    Cache.SessionPartition sessionPartition;  
  
    // Constructor of the controller for the Visualforce page.  
    public SessionPartitionController() {  
    }  
  
    // Adds counter value to the cache.  
    // This method is called when the Visualforce page loads.  
    public void init() {  
        // Create the partition instance based on the partition name  
        sessionPartition = getPartition();  
    }  
}
```



```

    }
}

// Returns the session partition based on the partition name
// given in the Visualforce page or the default value.
private Cache.SessionPartition getPartition() {
    if (sessionPartition == null) {
        sessionPartition = Cache.Session.getPartition(partitionInput);
    }

    return sessionPartition;
}

// Return counter from the cache.
public Integer getCounter() {
    return (Integer)getPartition().get(counterKeyInput);
}

// Invoked by the Submit button to save input values
// supplied by the user.
public PageReference save() {
    // Reset the initial key value in the cache
    getPartition().put(counterKeyInput, counterInitValue);

    return null;
}

// Method invoked by the Rerender button on the Visualforce page.
// Updates the values of various cached values.
// Increases the values of counter and the MyData counter if those
// cache values are still in the cache.
public PageReference go() {
    // Get the partition object
    sessionPartition = getPartition();
    // Increase the cached counter value or set it to 0
    // if it's not cached.
    if (sessionPartition.containsKey(counterKeyInput)) {
        sessionPartition.put(counterKeyInput, getCounter() + 1);
    } else {
        sessionPartition.put(counterKeyInput, counterInitValue);
    }

    return null;
}

// Method invoked by the Remove button on the Visualforce page.
// Removes the datetime cached value from the session cache.
public PageReference remove() {
    getPartition().remove(counterKeyInput);

    return null;
}

// Get and set methods for accessing variables
// that correspond to the input text fields on
// the Visualforce page.
public String getPartitionInput() {
    return partitionInput;
}

public String getCounterKeyInput() {
    return counterKeyInput;
}

public Integer getCounterInitValue() {
    return counterInitValue;
}


public void setPartitionInput(String partition) {
    this.partitionInput = partition;
}

```



```
}  
}
```

This is the Visualforce page that corresponds to the `SessionPartitionController` class.



```
<apex:page controller="SessionPartitionController" action="{!init}">  
  
  <apex:form >  
    <br/>Partition with Namespace Prefix: <apex:inputText value="{!partitionInput}"/>  
    <br/>Counter Key Name: <apex:inputText value="{!counterKeyInput}"/>  
    <br/>Counter Initial Value: <apex:inputText value="{!counterInitValue}"/>  
    <apex:commandButton action="{!save}" value="Save Key Input Values"/>  
  </apex:form>  
  
  <apex:outputPanel id="output">  
    <br/>Cached Counter: <apex:outputText value="{!counter}"/>  
  </apex:outputPanel>  
  
  <br/>  
  <apex:form >  
    <apex:commandButton id="go" action="{!go}" value="Rerender" rerender="output"/>  
    <apex:commandButton id="remove" action="{!remove}" value="Remove Key" rerender="ou</apex:form>  
  
</apex:page>
```

See Also

- [Apex Developer Guide: Platform Cache](#)

DID THIS ARTICLE SOLVE YOUR ISSUE?

Let us know so we can improve!

[Share your feedback](#)



DEVELOPER CENTERS

- [Heroku](#)
- [MuleSoft](#)
- [Tableau](#)
- [Commerce Cloud](#)
- [Lightning Design System](#)
- [Einstein](#)
- [Quip](#)

POPULAR RESOURCES

- [Documentation](#)
- [Component Library](#)
- [APIs](#)
- [Trailhead](#)
- [Sample Apps](#)
- [Podcasts](#)
- [AppExchange](#)

COMMUNITY

- [Trailblazer Community](#)
- [Events and Calendar](#)
- [Partner Community](#)
- [Blog](#)
- [Salesforce Admins](#)
- [Salesforce Architects](#)

