

### Simple Use Case

**Markup:**

```
<div id="myAutoComplete">
  <input id="myInput" type="text">
  <div id="myContainer"></div>
</div>
```

**Script:**

```
var myAutoComp = new YAHOO.widget.AutoComplete ("myInput",
  "myContainer", myDataSource);
```

Instantiates a new AutoComplete object, `myAutoComp`, which queries an existing DataSource `myDataSource`.

### Constructor

```
YAHOO.widget.AutoComplete(str | el ref input field, str |
  el ref suggestion container, obj DataSource instance[,
  obj configuration object]);
```

**Arguments:**

- (1) **HTML element (string or object):** Text input or textarea element.
- (2) **HTML element (string or object):** Suggestion container.
- (3) **DataSource instance (obj):** An instantiated DataSource object; see below for DataSource types and constructor syntax.
- (4) **Configuration object (object):** An optional object literal defines property values of an AutoComplete instance.

### Solutions

**Custom cell formatting:**

```
myAC.resultsTypeList = false; // pass data as an object
myAC.formatResult = function(oData, sQuery, sMatch) {
  return (sMatch + "(" + oData.param + ")");
}
```

**Custom local filtering:**

```
myAC.applyLocalFilter = true; // pass results thru filter
myAC.filterResults = function(sQuery, oFullResponse,
  oParsedResponse, oCallback) {
  var matches = [], matchee;
  for(var i=0; i<oParsedResponse.results.length; i++) {
    if(oParsedResponse.results[i].someValue > 0) {
      matches[matches.length] =
        oParsedResponse.results[i]
    }
  }
  oParsedResponse.results = matches;
  return oParsedResponse;
}
```

### Interesting Moments

Event	Arguments (passed via <i>args</i> array)
textboxFocusEvent/ textboxBlurEvent/ textboxChangeEvent	[0] AC instance
textboxKeyEvent	[0] AC instance; [1] keycode int
dataRequestEvent	[0] AC instance; [1] query string; [2] request object
dataReturnEvent	[0] AC instance; [1] query string; [2] results array
dataErrorEvent	[0] AC instance; [1] query string
containerExpandEvent/ containerCollapseEvent/ containerPopulateEvent	[0] AC instance
itemArrowToEvent/ itemArrowFromEvent	[0] AC instance; [1] <li> element
itemMouseOverEvent/ itemMouseOutEvent	[0] AC instance; [1] <li> element
itemSelectEvent	[0] AC instance; [1] <li> element; [2] item data object or array
selectionEnforceEvent	[0] AC instance
typeAheadEvent	[0] AC instance; [1] query string; [2] prefill string
unmatchedItemSelectEvent	[0] AC instance; [1] user selected string
Subscribe to AutoComplete Custom Events on your AutoComplete instance: <code>myAC.containerExpandEvent.subscribe(myFn[, myObj, bScope]);</code>	

### Abstract Methods

Method	Description
doBeforeLoadData	This overridable abstract method gives implementers access to the DataSource response before it is consumed by the AutoComplete instance and rendered into the results container.
doBeforeExpandContainer	This overridable abstract method gives implementers access to result data and DOM elements after the container has been rendered with results but before it is displayed to the user, for example to move the container to a different position on the screen.

### Dependencies

AutoComplete requires the YAHOO Global Object, Dom, and Event, and DataSource. Animation (for animated opening of the suggestion container) is optional.

YAHOO.widget.  
AutoComplete Key  
Properties:

---

- alwaysShow Container** (b)
- animHoriz** (b)
- animSpeed** (int)
- animVert** (b)
- applyLocalFilter** (b)
- autoHighlight** (b)
- delimChar** (char || array)
- forceSelection** (b)
- highlightClassName** (string)
- maxResultsDisplayed** (int)
- minQueryLength** (int)
- prehighlightClass Name** (string)
- queryDelay** (int)
- queryMatchCase** (b)
- queryMatchContains** (b)
- queryMatchSubset** (b)
- queryQuestionMark** (b)
- resultsTypeList** (b)
- suppressInputUpdate** (b)
- typeAhead** (b)
- typeAheadDelay** (int)
- useIframe** (b)
- useShadow** (b)