Map Module

Namespace: FSharp.Collections
Assembly: FSharp.Core.dll

Contains operations for working with values of type Map.

Functions and values

Function or value	Description
`Map.add key value table`	Returns a new map with the binding added to the given map. If a binding with the given key already exists in the input map, the existing binding is replaced by the new binding in the result map. ▶
`Map.change key f table`	Returns a new map with the value stored under key changed according to f. ▶
`Map.containsKey key table`	Tests if an element is in the domain of the map. ▶
`Map.count table`	The number of bindings in the map. ▶
`Map.empty`	The empty map. ▶
`Map.exists predicate table`	Returns true if the given predicate returns true for one of the bindings in the map.

Function or value	Description
	▶
`Map.filter predicate table`	Builds a new map containing only the bindings for which the given predicate returns 'true'. ▶
`Map.find key table`	Lookup an element in the map, raising `KeyNotFoundException` if no binding exists in the map. ▶
`Map.findKey predicate table`	Evaluates the function on each mapping in the collection. Returns the key for the first mapping where the function returns 'true'. Raise `KeyNotFoundException` if no such element exists. ▶
`Map.fold folder state table`	Folds over the bindings in the map ▶
`Map.foldBack folder table state`	Folds over the bindings in the map. ▶
`Map.forall predicate table`	Returns true if the given predicate returns true for all of the bindings in the map. ▶

Function or value	Description
`Map.isEmpty table`	Is the map empty? ▶
`Map.iter action table`	Applies the given function to each binding in the dictionary ▶
`Map.keys table`	The keys in the map. The sequence will be ordered by the keys of the map. ▶
`Map.map mapping table`	Builds a new collection whose elements are the results of applying the given function to each of the elements of the collection. The key passed to the function indicates the key of element being transformed. ▶
`Map.maxKeyValue table`	Returns binding for the largest key in the map. Raise `KeyNotFoundException` when map is empty. ▶
`Map.minKeyValue table`	Returns binding for the smallest key in the map. Raise `KeyNotFoundException` when map is empty. ▶

Function or value	Description	
`Map.ofArray elements`	Returns a new map made from the given bindings. ▶	
`Map.ofList elements`	Returns a new map made from the given bindings. ▶	
`Map.ofSeq elements`	Returns a new map made from the given bindings. ▶	
`Map.partition predicate table`	Builds two new maps, one containing the bindings for which the given predicate returns 'true', and the other the remaining bindings. ▶	
`Map.pick chooser table`	Searches the map looking for the first element where the given function returns a `Some` value. Raise `KeyNotFoundException` if no such element exists. ▶	
`Map.remove key table`	Removes an element from the domain of the map. No exception is raised if the element is not present. ▶	
`Map.toArray table`	Returns an array of all key-value pairs in the mapping. The array will be ordered by the keys of the map. ▶	
`Map.toList table`	Returns a list of all key-value pairs in the mapping. The list will be ordered by the keys of the map.	

Function or value	Description	
	▶	
`Map.toSeq table`	Views the collection as an enumerable sequence of pairs. The sequence will be ordered by the keys of the map. ▶	
`Map.tryFind key table`	Lookup an element in the map, returning a `Some` value if the element is in the domain of the map and `None` if not. ▶	
`Map.tryFindKey predicate table`	Returns the key of the first mapping in the collection that satisfies the given predicate. Returns 'None' if no such element exists. ▶	
`Map.tryPick chooser table`	Searches the map looking for the first element where the given function returns a `Some` value. ▶	
`Map.values table`	The values in the map, including the duplicates. The sequence will be ordered by the keys of the map. ▶