# Set Module

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

Contains operations for working with values of type Set.

## Functions and values

| Function or value | Description | |
|---|---|---|
| `Set.add value set` | Returns a new set with an element added to the set. No exception is raised if the set already contains the given element. ▶ | XML MD |
| `Set.contains element set` | Evaluates to "true" if the given element is in the given set. ▶ | XML MD |
| `Set.count set` | Returns the number of elements in the set. Same as `size`. ▶ | XML MD |
| `Set.difference set1 set2` | Returns a new set with the elements of the second set removed from the first. ▶ | XML MD |
| `Set.empty` | The empty set for the type 'T. ▶ | XML MD |
| `Set.exists predicate set` | Tests if any element of the collection satisfies the given predicate. If the input function is `predicate` and the elements are `i0...iN` then computes `p i0 or ... or p iN`. | XML MD |

| Function or value | Description |
|---|---|
| | ▶ |
| `Set.filter predicate set` | Returns a new collection containing only the elements of the collection for which the given predicate returns True. ▶ |
| `Set.fold folder state set` | Applies the given accumulating function to all the elements of the set ▶ |
| `Set.foldBack folder set state` | Applies the given accumulating function to all the elements of the set. ▶ |
| `Set.forall predicate set` | Tests if all elements of the collection satisfy the given predicate. If the input function is `f` and the elements are `i0...iN` and "j0...jN" then computes `p i0 && ... && p iN`. ▶ |
| `Set.intersect set1 set2` | Computes the intersection of the two sets. ▶ |
| `Set.intersectMany sets` | Computes the intersection of a sequence of sets. The sequence must be non-empty. ▶ |
| `Set.isEmpty set` | Returns "true" if the set is empty. ▶ |

| Function or value | Description | |
|---|---|---|
| `Set.isProperSubset set1 set2` | Evaluates to "true" if all elements of the first set are in the second, and at least one element of the second is not in the first.<br>▶ | |
| `Set.isProperSuperset set1 set2` | Evaluates to "true" if all elements of the second set are in the first, and at least one element of the first is not in the second.<br>▶ | |
| `Set.isSubset set1 set2` | Evaluates to "true" if all elements of the first set are in the second<br>▶ | |
| `Set.isSuperset set1 set2` | Evaluates to "true" if all elements of the second set are in the first.<br>▶ | |
| `Set.iter action set` | Applies the given function to each element of the set, in order according to the comparison function.<br>▶ | |
| `Set.map mapping set` | Returns a new collection containing the results of applying the given function to each element of the input set.<br>▶ | |
| `Set.maxElement set` | Returns the highest element in the set according to the ordering being used for the set.<br>▶ | |

| Function or value | Description |
|---|---|
| `Set.minElement set` | Returns the lowest element in the set according to the ordering being used for the set. ▶ |
| `Set.ofArray array` | Builds a set that contains the same elements as the given array. ▶ |
| `Set.ofList elements` | Builds a set that contains the same elements as the given list. ▶ |
| `Set.ofSeq elements` | Builds a new collection from the given enumerable object. ▶ |
| `Set.partition predicate set` | Splits the set into two sets containing the elements for which the given predicate returns true and false respectively. ▶ |

| Function or value | Description | |
|---|---|---|
| `Set.remove value set` | Returns a new set with the given element removed. No exception is raised if the set doesn't contain the given element.<br>▶ | |
| `Set.singleton value` | The set containing the given element.<br>▶ | |
| `Set.toArray set` | Builds an array that contains the elements of the set in order.<br>▶ | |
| `Set.toList set` | Builds a list that contains the elements of the set in order.<br>▶ | |
| `Set.toSeq set` | Returns an ordered view of the collection as an enumerable object.<br>▶ | |
| `Set.union set1 set2` | Computes the union of the two sets.<br>▶ | |
| `Set.unionMany sets` | Computes the union of a sequence of sets.<br>▶ | |