```
db.collection.createIndex( { <location field> : "2d" } ,
                           { min : <lower bound> , max : <upper bound> } )
```

Define Location Precision for a `2d` Index

By default, a `2d` index on legacy coordinate pairs uses 26 bits of precision, which is roughly equivalent to 2 feet or 60 centimeters of precision using the default range of -180 to 180. Precision is measured by the size in bits of the *geohash* values used to store location data. You can configure geospatial indexes with up to 32 bits of precision.

Index precision does not affect query accuracy. The actual grid coordinates are always used in the final query processing. Advantages to lower precision are a lower processing overhead for insert operations and use of less space. An advantage to higher precision is that queries scan smaller portions of the index to return results.

To configure a location precision other than the default, use the `bits` option when creating the index. Use following syntax:

```
db.<collection>.createIndex( {<location field> : "<index type>"} ,
                             { bits : <bit precision> } )
```

For information on the internals of geohash values, see *Calculation of Geohash Values for 2d Indexes* (page 499).

Query a `2d` Index

The following sections describe queries supported by the `2d` index.

Points within a Shape Defined on a Flat Surface

To select all legacy coordinate pairs found within a given shape on a flat surface, use the `$geoWithin` operator along with a shape operator. Use the following syntax:

```
db.<collection>.find( { <location field> :
                         { $geoWithin :
                            { $box|$polygon|$center : <coordinates>
                      } } } )
```

The following queries for documents within a rectangle defined by `[ 0 , 0 ]` at the bottom left corner and by `[ 100 , 100 ]` at the top right corner.

```
db.places.find( { loc :
                  { $geoWithin :
                     { $box : [ [ 0 , 0 ] ,
                                [ 100 , 100 ] ]
                 } } } )
```

The following queries for documents that are within the circle centered on `[ -74 , 40.74 ]` and with a radius of `10`:

```
db.places.find( { loc: { $geoWithin :
                          { $center : [ [-74, 40.74 ] , 10 ]
                } } } )
```

For syntax and examples for each shape, see the following:

- `$box`
- `$polygon`