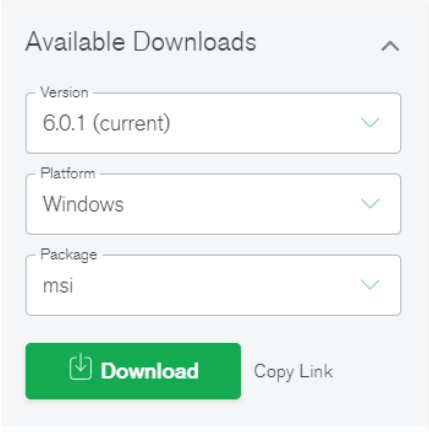
MongoDB的安装、启动、连接和CRUD操作初步

安装、启动、连接MongoDB（windows）

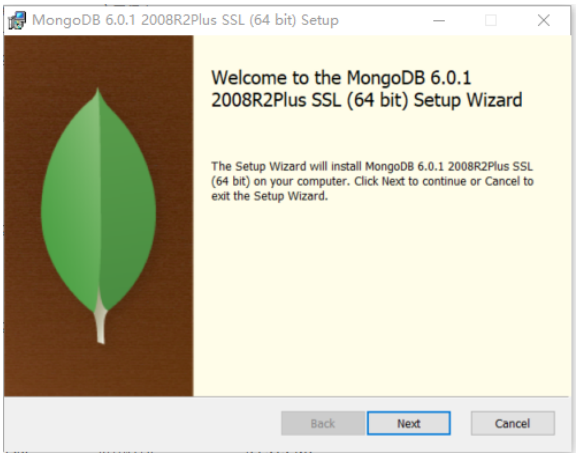
- 官方文档：https://www.mongodb.com/docs/manual/tutorial/install-mongodb-on-windows/
- 安装包官方地址：https://www.mongodb.com/try/download/community

进入安装包官方地址[:https://www.mongodb.com/try/download/community]，如下图所示，在页面右侧选择下载合适的安装包，点击Download下载。

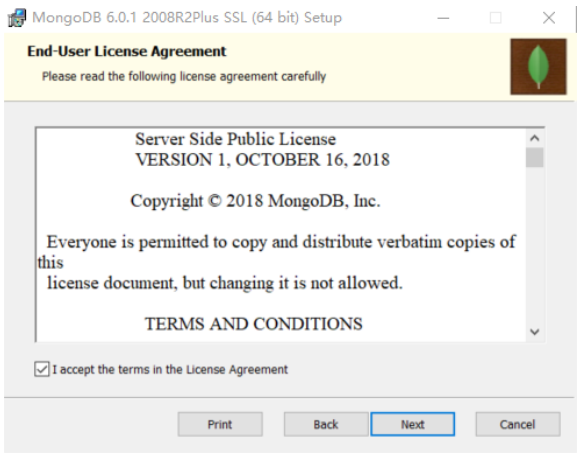


下载完毕后，双击安装包，开始安装

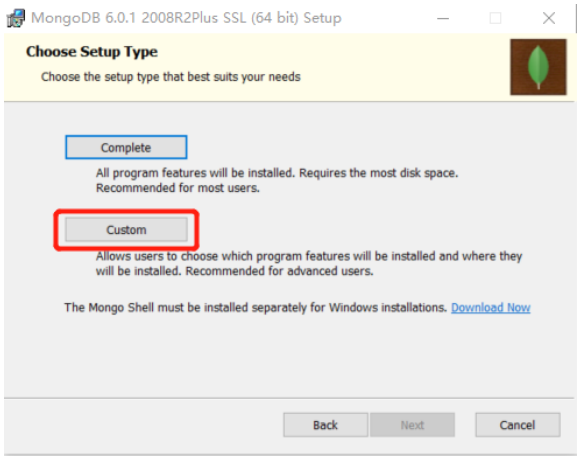
step1: 点击Next



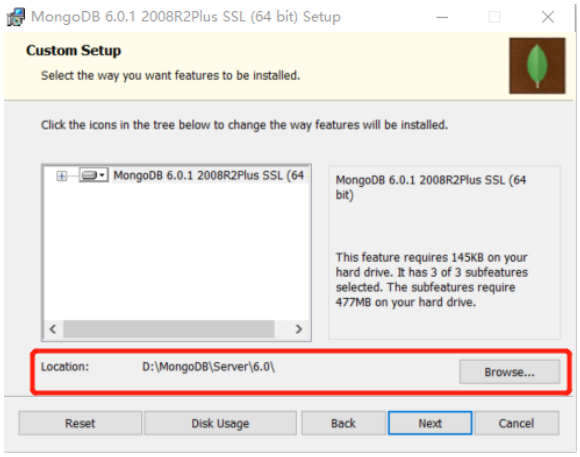
step2: 勾选后点击Next



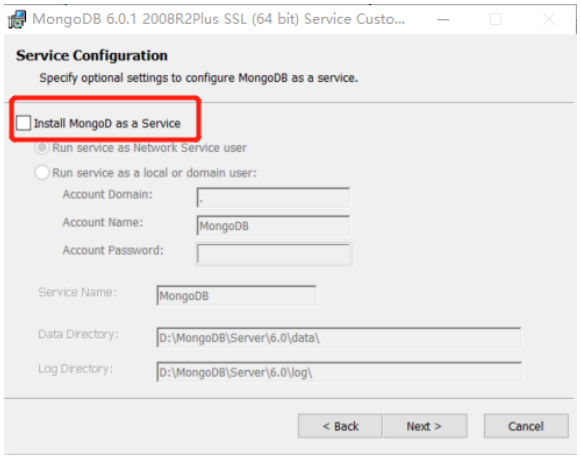
step3: 选择Custom模式，Complete模式比较大且默认安装在C盘无法选择，没必要



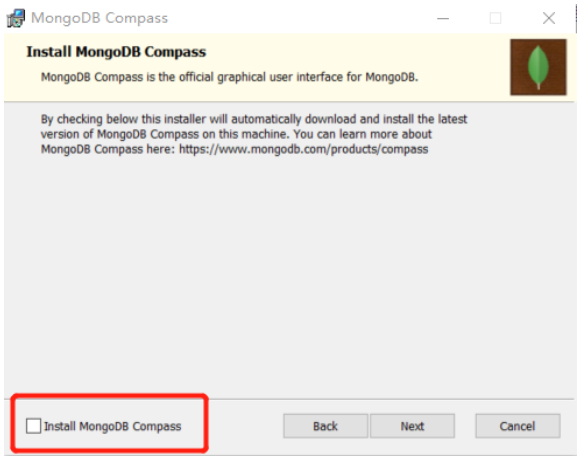
step4: 选择安装路径，安装组件按照默认就好



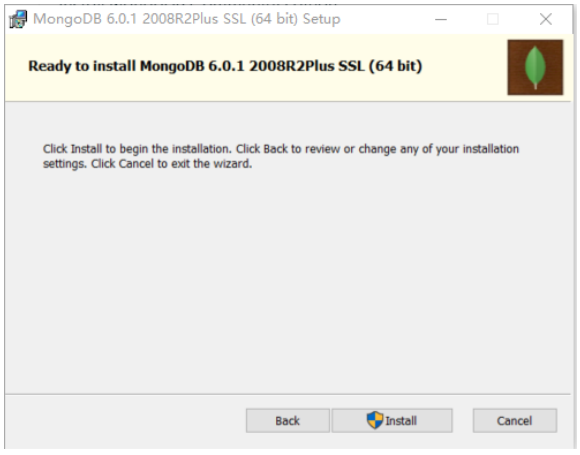
step5: 选择服务配置，取消勾选Install MongoD as a Service，采用本地mongodb模式。点击next



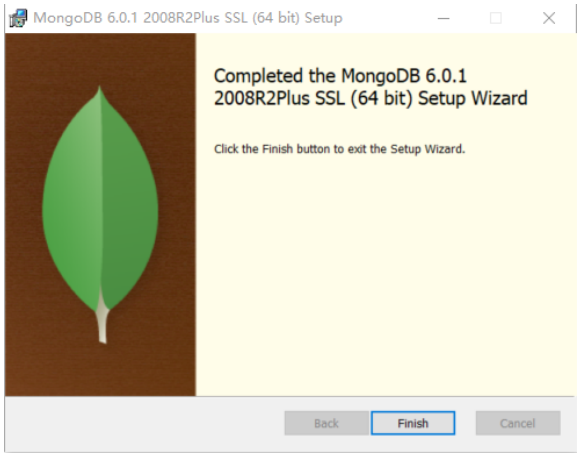
step6: 取消勾选Install MongoDB Compass，这是一个可视化界面，用不到。



step7: 点击Install



step8: 安装完成，点击Finish



上述是MongoDB安装过程，下面需要进行启动与连接操作。

首先配置环境变量，使CMD能够识别MongoDB相关的命令。

打开我的电脑->属性->高级系统设置->环境变量->系统变量->Path->新建->添加MongoDB下bin目录的位置->点击确定。
我的MongoDB的bin目录位置是 D:\MongoDB\Server\6.0\bin\ 注意最后的“\”一定不能省略。

之后，为本次实验MongoDB数据库创建文件夹，确保实验过程中存放的数据在我们想存放的位置。
例如，创建"E:\Software_Data\MongoDB\data\testdb"目录，后续的实验MongoDB将数据存放在testdb文件夹下。

配置环境变量并创建数据库存放文件夹后，我们开始初始化、启动、连接数据库。

以管理员模式打开cmd（第一次初始化需要管理员模式，后续无所谓）。
输入下述命令：

```
mongod --dbpath="E:\Software_Data\MongoDB\data\testdb"
```

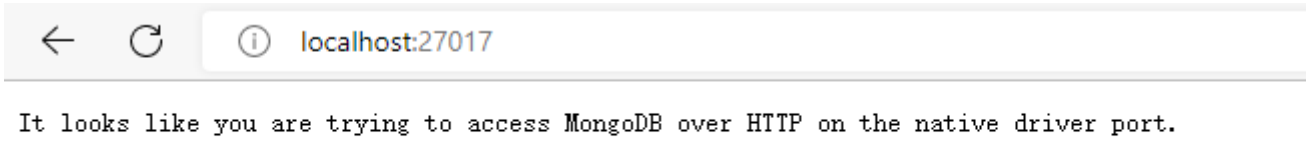
初始化数据库，并启动mongodb服务等待链连接。其中后面的地址"E:...\data\testdb"，是本次启动的数据库所在的地址，可以自行指定。第一次启动时会初始化数据库，因此时间稍慢一些。
当光标不再移动，最后两行出现下述内容时，认为数据库启动成功，可以继续进行连接操作：

```
"ctx":"listener","msg":"Listening on","attr":{"address":"127.0.0.1"}}
"ctx":"listener","msg":"Waiting for connections","attr":{"port":27017,"ssl":"off"}}
```

打开浏览器，在浏览器中输入：

```
http://localhost:27017/
```

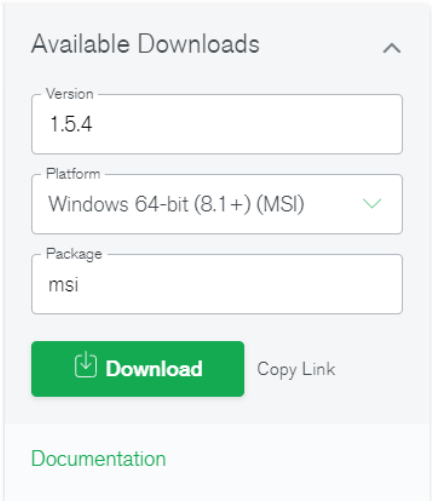
如果出现下述结果，则mongodb初始化、启动、连接功能测试完毕。



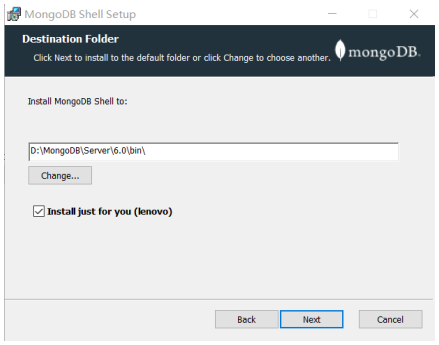
由于mongodb6.0版本下，bin文件夹内不包含mongo.exe执行文件（更早的版本包含），因此需要安装mongo shell组件连接mongod服务，以操作数据库。下面进行mongo shell的安装、配置与连接。

- 官方文档：https://www.mongodb.com/docs/mongodb-shell/
- 安装包官方地址：https://www.mongodb.com/try/download/shell?jmp=docs

打开安装包官方地址，按照下图选择安装包的类型，点击Download。



下载完毕，双击安装包，点击next，到输入安装地址时注意，可以安装到上文安装的MongoDB软件的bin目录下，如下图所示，这样我们就不需要配置mongo shell的环境变量。



打开cmd，输入mongosh --version。如果输出1.5.4版本号，则表明mongosh安装配置成功。

接下来启动mongodb，并使用mongosh连接mongod服务。

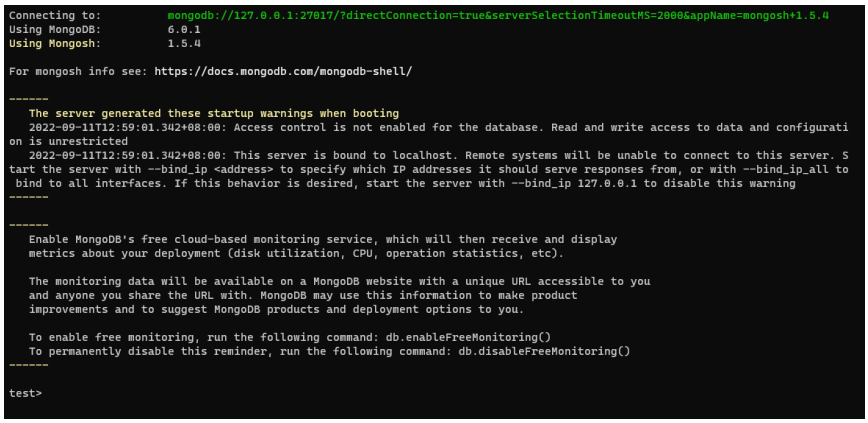
打开cmd，输入下述命令启动数据库

```
mongod --dbpath="E:\Software_Data\MongoDB\data\testdb"
```

在打开另一个cmd，输入下述命令，启动mongosh，并连接mongod服务。

```
mongosh
```

上述命令默认连接地址为localhost、端口为27017的mongod服务，当出现下输入结果时，可认为连接成功。



后续可以在mongosh里面进行CRUD操作。

安装、启动、连接MongoDB（mac）

参考文档：https://www.jianshu.com/p/1fc68690a747 （亲测有效）

按照上述文档进行操作，可能会出现一些问题，下面列出了常见的两个问题及解决方案：

1.

输入命令

```
mongod --dbpath /usr/local/var/mongodb --logpath /usr/local/var/log/mongodb/mongo.log --fork
```

如果出现类似下述的错误：

```
about to fork child process, waiting until server is ready for connections.
forked process: 44890
ERROR: child process failed, exited with 48
To see additional information in this output, start without the "--fork" option.
```

可以尝试把/usr/local/var/mongodb/mongod.lock和/usr/local/var/mongodb/diagnostic.data
这个两个文件删除

2.

安装成功之后，如果输入 `mongo` 后出现 `command not found` 。
这可能是有由于下载的安装包版本问题导致/usr/local/MongoDB/bin/文件夹中没有mongo这个可执行文件，将助教提供的可执行文件拷贝到这个文件夹中即可。

mongo.exe网盘下载：
链接：https://pan.baidu.com/s/1BsvV0qpoSZxSMjx0E1V7AA
提取码：eweh

CRUD操作初步

本文档只列出基本的增删改查操作示例，完整操作命令可以参考下述两个资料：

- 官方文档：https://www.mongodb.com/docs/manual/crud/
- RUNOOB：https://www.runoob.com/mongodb/mongodb-tutorial.html

1.创建数据库

在mongosh中输入下述命令：

```
use school
```

会发现，命令行变成了"school>"样式。

表示使用school数据库进行操作，如果school数据库不存在则自动创建。后续的CRUD操作，我们将在school数据库中进行操作。

2.插入

MongoDB常用插入命令：

```
db.COLLECTION_NAME.insertOne()
db.COLLECTION_NAME.insertMany()
```

插入示例：

```
db.student.insertOne({"name":"zhang san", "year":2020, "age":20})
db.student.insertMany([{"name":"zhang san","year":2021, "age":18},
                       {"name":"liu er", "year":2020, "age":21},
                       {"name":"li si","year":2021, "age":19},
                       {"name":"wang wu","year":2022, "age":18}])
```

第一条插入命令一次插入一条记录，第二条插入命令一次可插入一个集合。注意，集合需要用"[...]"包含，并且集合内记录间需要用","分隔开。

如果插入成功会分别显示结果：

```
{
  acknowledged: true,
  insertedId: ObjectId("631d55faab43aecd2548b4c8")
}

{
  acknowledged: true,
  insertedIds: {
    '0': ObjectId("631d563eab43aecd2548b4c9"),
    '1': ObjectId("631d563eab43aecd2548b4ca"),
    '2': ObjectId("631d563eab43aecd2548b4cb"),
    '3': ObjectId("631d563eab43aecd2548b4cc")
  }
}
```

在mongodb中，当向集合插入数据时，如果该集合不存在，则会自动创建；如果插入数据不显示包含"_id"字段，则系统会自动分配。

3.查询

查询的语法格式如下所示，该结构为一个嵌套结构，的内容任然可以是一个{:,...}结构。

```
db.COLLECTION_NAME.find({<field1>:<value1>,...})
```

先考虑一个查询条件为空的查询，结果应包含之前插入的5条学生信息。注意，记录中多出了"_id"字段，为系统自动分配：

```
db.student.find({})
```

再考虑一个查询，查询名字是"zhang san"的同学信息，结果应返回两条"name"为"zhang san"的记录：

```
db.student.find({"name":"zhang san"})
```

如果想进一步限定查询范围，那就需要增加限定条件。比如，查询名字是"zhang san"，且在2021年入学的同学，此时，结果相对上一次查询，仅剩一条记录：

```
db.student.find({"name":"zhang san", "year":2021})
```

上述并列的两个查询条件，默认是"and"的关系。如果需要表达"or"关系，例如查询2021年入学，或者年龄为20岁的同学信息，结果应包含三条记录：

```
db.student.find({$or:[{"year":2021}, {"age":20}]})
```

利用"[...]"将条件作为一个集合罗列出来，利用"%or"运算符表达集合中的条件是"或"的关系。
除了"and""or"关系，mongodb还提供了表达数量关系的操作符，例如查询入学年份在2021年之前的同学，结果应包含两条2020年入学的学生记录：

```
db.student.find({"year":{$lt:2021}})
```

其中"$lt"表示小于关系，{$lt:2021}表示"小于2021"。根据上述示例可知，mongodb也提供了下述语法结构：

```
{<operator>:<value>}
```

mongodb提供了等于、小于、大于、小于等于、存在于（$in）等多种关系操作符，细节请参考文档。

最后来一个比较全面的查询语句，查询名字为"zhang san"，且入学年份在2021之前或年龄为18岁的学生，结果应返回两条记录。

```
db.student.find({"name":"zhang san", $or:[{"year":{$lt:2021}}, {"age":18}]})
```

4.更新

db.collection.updateOne()只更新符合条件的第一个文档
db.collection.updateMany() 更新所有符合条件的文档

更新语句的语法结构如下：

```
db.COLLECTION_NAME.updateMany(<query>,<update>)
```

更新语句首先根据"<query>"查询出符合条件的记录用作更新，之后根据"<update>"语句更新记录。

例如，将所有名字为"zhang san"的同学，入学年份更改为2022年：

```
db.student.updateMany({"name":"zhang san"},
                      {$set:{"year":2022}})
```

利用"db.student.find({})"查询结果。

将所有入学年份为2022年的同学，年龄增加一岁：

```
db.student.updateMany({"year":2022},
                      {$inc:{"age":1}})
```

利用"db.student.find({})"查询结果。

5.删除

删除语句于更新语句类似。
db.collection.deleteOne()只更新符合条件的第一个文档
db.collection.deleteMany() 更新所有符合条件的文档

删除语句的语法结构如下：

```
db.COLLECTION_NAME.deleteMany(<query>)
```

删除语句会根据"<query>"，删除符合查询条件的记录。

例如，删除名字带有"zhang san"的同学，仅删除第一条符合条件的文档：

```
db.student.deleteOne({"name":"zhang san"})
```

利用"db.student.find({})"查询结果。

删除入学年份为2022的全部同学的信息：

```
db.student.deleteMany({"year":2022})
```

利用"db.student.find({})"查询结果。

删除student集合下全部文档：

```
db.student.deleteMany({})
```