Matt Weaver

final project documentation-server

C #

**Purpose:** the purpose of this program is to create user interface that talks to a server and stores information in a database. The information stored it information about DVDs

**Classes:** This program has eight classes, seven which our support classes, they are DVD, genre, utilities, server actions, session, log writer. The eight class which is server Is the main program class.

**Note:** the entire project also contains a client program which is mentioned here, but not in great detail for more information about the server program see the final project server documentation. Also due to the scope of this sentiment the main methods will be discussed as the main pseudocode instead of class documentation. However, the class documentation will so contain the names of the methods

**files needed**: there is a database file located in the bin folder in the debugging folder of this project. The database is called DVDB\_001\_DVD\_master

**Class diagram**

|  |
| --- |
| **DVD** |
| -DVD\_action: string  -DVD\_ID: string  -DVD\_title: string  -DVD\_first: string  -DVD\_last: string  -DVD\_first2: string  -DVD\_last2: string  -DVD\_cp: string  -DVD\_genre: short  -DVD\_rt: short  -DVD\_type: char  -DVD\_stat: char  -number validation: regular expression (@"^\d+$")  -character validation: regular expression (@"[^a-zA-Z]") |
| + DVD ()-constructor  + DVD (ID: string, title: string, first: string, last: string, first2: string, last2: string, company: string, genre: short, runtime: short, type: char, status: char)-constructor  +getAction (): string  + setAction (action: string): void  + getID ():string  + setID(ID:string):void  + getTitle ():string  + setTitle (title: string): void  +getFirstname():string  +setFirstname(first:string):void  +getLastname (): string  + setLastname (last: string): void  + +getFirstname2():string  +setFirstname2(first2:string):void  +getLastname2 (): string  + setLastname2 (last2: string): void  +getCompany ():string  +setCompany (company:string):void  +getGenre (): short  +setGenre (genre: short):void  +getRuntime (): short  +setRuntime (runtime: short):void  +getType (): char  +setType (type: char):void  +getStatus (): char  +setStatus (status: char):void |

**Method descriptions**

1. DVD-constructor

* set the variables to the following values
* DVD\_ID = 0;
* DVD\_title = Title
* DVD\_first = First
* DVD\_last = Last
* DVD\_first2 = First2
* DVD\_last2 = Last2
* DVD\_cp = Company
* DVD\_genre = 0
* DVD\_rt = 0
* DVD\_type = U
* DVD\_stat = E

1. DVD (DVD (ID: string, title: string, first: string, last: string, first2: string, last2: string, company: string, genre: short, runtime: short, type: char, status: char)-constructor

* set the setters to the appropriate parameter value

1. class getters

* for each getter return the appropriate private variable

1. SetID

* if ID length is greater than zero then
* DVD\_ID = ID
* else if DVD\_ID = null
* DVD\_ID = "0"
* else DVD\_ID = "0"

1. SetTitle

* if title length is greater than zero and less than or equal to 50 then
* DVD\_title = title
* else if DVD\_title =null
* DVD\_title = "Title"
* else DVD\_title = "Title"

1. SetFirstname

* if first length is greater than zero and less than or equal to 30 then
* DVD\_first = first
* else if DVD\_first =null
* DVD\_first = "First"
* else DVD\_first = "First"

1. SetLastname

* if last length is greater than zero and less than or equal to 30 then
* DVD\_last = last
* else if DVD\_last =null
* DVD\_last = "last"
* else DVD\_last = "Last"

1. SetFirstname2

* if first2 length is greater than zero and less than or equal to 30 then
* DVD\_first2 = first2
* else if first2 =null
* DVD\_first2 = "First2"
* else DVD\_first2 = "First2"

1. SetLastname2

* if last2 length is greater than zero and less than or equal to 30 then
* DVD\_last2 = last 2
* else if DVD\_last2=null
* DVD\_last2 = "Last2"
* else DVD\_last2 = "Last2

1. SetCompany

* if company length is greater than zero and less than or equal to 40 then
* DVD\_cp= company
* else if DVD\_cp=null
* DVD\_cp= " Company"
* else DVD\_cp= " Company"

1. SetRuntime

* if runtime is greater than zero and less than or equal to 300 then
* DVD\_rt = runtime
* else DVD\_rt = 0

1. SetGenre

* if genre is greater than or equal to zero and less than 30 then
* DVD\_genre = genre
* else DVD\_genre = 0

1. SetType

* if type =D or B or U then
* DVD\_type = type
* else DVD\_type =U

1. SetStatus

* if status =E or N then
* DVD\_stat = status
* else DVD\_stat = E

**class diagram**

|  |
| --- |
| **Genre** |
| **-**DVD\_Genre\_ID: short  -DVD\_Genre\_Descript: string |
| + Genre()-constructor  + getGenreID (): short  + getGenreDescription(): String  + setGenreID (genreID: short): void  + setGenreDescription(description: string): void |

**Method descriptions**

1. Genre-constructor

* DVD\_Genre\_ID = 0
* DVD\_Genre\_Descript = unknown

1. getGenreID

* return DVD\_Genre\_ID

1. setGenreID

* if genre ID is greater than or equal to zero and less than to 30
* DVD\_Genre\_ID = genreID
* else DVD\_Genre\_ID = 0

1. getGenreDescription

* return DVD\_Genre\_Descript

1. setGenreID

* if description is greater than zero and less than or equal to 20
* DVD\_Genre\_Descript = description
* else DVD\_Genre\_Descript = unknown

**class diagram**

|  |
| --- |
| **Utilities** |
| + static GetNodeText (xpath: string, XML: string): string |

**Method descriptions**

1. GetNodeText

* declare a new XML document called XML doc set it equal to a new empty XML document
* declare a string called value and set it equal to or empty
* open try
* load the XML string into the XML document
* set value equal to GetNodeText with the X path and the XML document as parameters.
* Close try
* catch any appropriate exceptions
* clear the screen for each exception
* let the user know what the error is
* write the exception to the session class using the log message method
* set value equal to null
* do the last three steps for each exception
* return value

**class diagram**

|  |
| --- |
| **log writer** |
| -static writer:streamwriter  -static Logwriter:logwriter |
| -Logwriter ()-constructor  + static instance (): log writer  + WriteLogMessage (message: string): void |

**Method descriptions**

1. log writer-constructor

* writer = new stream writer with name of log and true as parameters. The name of the log is up to the programmer's discretion just make sure it's in the server source folder.
* Set the auto flush property of the writer to true

1. instance

* if log writer equals null then
* create a new empty log writer and set it equal to the log writer variable
* return log writer variable

1. WriteLogmessage

* use the writer with the right line method to write the message

**class diagram**

|  |
| --- |
| **database** |
| -ID: string  -new ID: string  -title: string  -first: string  -last: string  -first2: string  -last2: string  -company: string  -genre: short  -runtime: short  -type: char  -status: char |
| + Genre list (): genre [ ]  + addall (title: string, first: string, last: string, first2: string, last2: string, company: string, genre: short runtime: short, type: char, status: char): long  + add title (title: string, status: char): long  + delete (DVD\_ID: long): void  + purge (DVD\_ID: long): void  + undelete (DVD\_ID: long): void  + search actor1 (last: string): DVD [ ]  + search actor2(last2: string): DVD [ ]  + search company (company: string): DVD [ ]  + search genre (genre: short): DVD [ ]  + search runtime (runtime: short): DVD [ ]  + search type (type: char) : DVD [ ]  + search status (status: char): DVD [ ]  + search title (title: string): DVD [ ]  + select ID (DVD\_ID: long): OLEDBdatareader  +select Title (DVD\_title: string): OLEDBdatareader  +updateall (DVD\_ID: long, title: string, first: string, last: string, first2: string, last2: string, company: string, genre: short runtime: short, type: char, status: char): void  + update actor 1(DVD\_ID, first: string, last: string): void  + update actor 21(DVD\_ID, first2: string, last2: string): void  + update company (DVD\_ID, company: string): void  + update genre (DVD\_ID: long, genre: short): void  + update runtime (DVD\_ID: long, runtime: short): void  + update title (DVD\_ID: long, title: string): void  + update type (DVD\_ID: long, type: char): void |

**Database methods pseudocode**

1. **add all**
2. declare a string called strSQL set it equal to empty
3. declare long integer called DVD number set it equal to zero
4. declare a string called connstring set it equal to the correct string to connect the database
5. create a new database connection called cn set it equal to new database connection with the connection string as a parameter
6. log the message of the connection string path to the session
7. use the following information to insert information into the DVDT\_001\_DVD\_master in the database using SQL insert. Make sure to assign the SQL insert string to str SQL. The column name, size and datatype is information from the database. The value is the value the user gives

|  |  |  |  |
| --- | --- | --- | --- |
| Column name | Value | Datatype | Size |
| DVD\_Title | @DVD\_Title | Char | 50 |
| DVD\_First | @DVD\_First | Char | 30 |
| DVD\_Last | @DVD\_Last | Char | 30 |
| DVD\_First2 | @DVD\_First2 | Char | 30 |
| DVD\_Last2 | @DVD\_Last2 | Char | 30 |
| DVD\_Company | @DVD\_Company | Char | 40 |
| DVD\_Genre | @DVD\_Genre | Small int |  |
| DVD\_Runtime | @DVD\_Runtime | Small int |  |
| DVD\_Type | @DVD\_Type | Char | 1 |
| DVD\_Stat | @DVD\_Stat | Char | 1 |

1. Declare a new database command and set it equal to a database command with with the SQL string and the database connection as parameters
2. do the next four steps for each parameter listed in the method. In the order which they are listed
3. Declare a new database parameter with the value, and datatype and size listed above as parameters
4. make the parameter direction input
5. make the parameter value equal to the parameter listed in the declaration of the method in the order in which they are listed
6. use database command parameters method to add the parameter
7. once the last parameter has been added open the connection to the database
8. execute the non-query command
9. set the SQL string to the select identity command
10. set the database command to a new database command with the SQL string and database connection as parameters
11. set DVD number equal to the database command execute scalar, converted to a 64-bit integer which is a long
12. close the connection to the database
13. return DVD number
14. **add title**
15. declare a string called strSQL set it equal to empty
16. declare long integer called DVD number set it equal to zero
17. declare a string called connstring set it equal to the correct string to connect the database
18. create a new database connection called cn set it equal to new database connection with the connection string as a parameter
19. log the message of the connection string path to the session
20. use the following information to insert information into the DVDT\_001\_DVD\_master in the database using SQL insert. Make sure to assign the SQL insert string to str SQL. The column name, size and datatype is information from the database. The value is the value the user gives

|  |  |  |  |
| --- | --- | --- | --- |
| Column name | Value | Datatype | Size |
| DVD\_Title | @DVD\_Title | Char | 50 |
| DVD\_Stat | @DVD\_Stat | Char | 1 |

1. Declare a new database command and set it equal to a database command with with the SQL string and the database connection as parameters
2. do the next four steps for each parameter listed in the method. In the order which they are listed
3. Declare a new database parameter with the value, and datatype and size listed above as parameters
4. make the parameter direction input
5. make the parameter value equal to the parameter listed in the declaration of the method in the order in which they are listed
6. use database command parameters method to add the parameter
7. once the last parameter has been added open the connection to the database
8. execute the non-query command
9. set the SQL string to the select identity command
10. set the database command to a new database command with the SQL string and database connection as parameters
11. set DVD number equal to the database command execute scalar, converted to a 64-bit integer which is a long
12. close the connection to the database
13. return DVD number
14. **delete**
15. declare a string called strSQL set it equal to empty
16. declare a string called connstring set it equal to the correct string to connect the database
17. create a new database connection called cn set it equal to new database connection with the connection string as a parameter
18. log the message of the connection string path to the session
19. use the following information to update information in the DVDT\_001\_DVD\_master in the database using SQL update. Make sure to assign the SQL update string to str SQL. The column name, and datatype is information from the database. If the value has @in front of it is given by the user. If not change the indicated column to that value The where column below indicates when to change the value

|  |  |  |  |
| --- | --- | --- | --- |
| Column name | Value | Datatype | Where |
| DVD\_ID | @DVD\_ID | Big int | DVD\_ID =@DVD\_ID |
| DVD\_Stat | N |  |  |

1. Declare a new database command and set it equal to a database command with the SQL string and the database connection as parameters
2. do the next four steps for each parameter listed in the method. In the order which they are listed
3. Declare a new database parameter with the value, and datatype and size listed above as parameters
4. make the parameter direction input
5. make the parameter value equal to the parameter listed in the declaration of the method in the order in which they are listed
6. user database command parameters method to add the parameter
7. once the last parameter has been added open the connection to the database
8. execute the non-query command
9. close the connection to the database
10. **undelete**
11. declare a string called strSQL set it equal to empty
12. declare a string called connstring set it equal to the correct string to connect the database
13. create a new database connection called cn set it equal to new database connection with the connection string as a parameter
14. log the message of the connection string path to the session
15. use the following information to update information in the DVDT\_001\_DVD\_master in the database using SQL update. Make sure to assign the SQL update string to str SQL. The column name, and datatype is information from the database. If the value has @in front of it is given by the user. If not change the indicated column to that value The where column below indicates when to change the value

|  |  |  |  |
| --- | --- | --- | --- |
| Column name | Value | Datatype | Where |
| DVD\_ID | @DVD\_ID | Big int | DVD\_ID =@DVD\_ID |
| DVD\_Stat | E |  |  |

1. Declare a new database command and set it equal to a database command with the SQL string and the database connection as parameters
2. do the next four steps for each parameter listed in the method. In the order which they are listed
3. Declare a new database parameter with the value, and datatype and size listed above as parameters
4. make the parameter direction input
5. make the parameter value equal to the parameter listed in the declaration of the method in the order in which they are listed
6. user database command parameters method to add the parameter
7. once the last parameter has been added open the connection to the database
8. execute the non-query command
9. close the connection to the database
10. **purge**
11. declare a string called strSQL set it equal to empty
12. declare a string called connstring set it equal to the correct string to connect the database
13. create a new database connection called cn set it equal to new database connection with the connection string as a parameter
14. log the message of the connection string path to the session
15. use the following information to delete the information in the DVDT\_001\_DVD\_master in the database using SQL delete. Make sure to assign the SQL delete string to str SQL. The column name, and datatype is information from the database. If the value has @in front of it is given by the user. If not change the indicated column to that value The where column below indicates when to change the value

|  |  |  |  |
| --- | --- | --- | --- |
| Column name | Value | Datatype | Where |
| DVD\_ID | @DVD\_ID | Big int | DVD\_ID =@DVD\_ID and DVD\_stat = N |

1. Declare a new database command and set it equal to a database command with the SQL string and the database connection as parameters
2. do the next four steps for each parameter listed in the method. In the order which they are listed
3. Declare a new database parameter with the value, and datatype and size listed above as parameters
4. make the parameter direction input
5. make the parameter value equal to the parameter listed in the declaration of the method in the order in which they are listed
6. user database command parameters method to add the parameter
7. once the last parameter has been added open the connection to the database
8. execute the non-query command
9. close the connection to the database
10. **update all**
11. declare a string called strSQL set it equal to empty
12. declare a string called connstring set it equal to the correct string to connect the database
13. create a new database connection called cn set it equal to new database connection with the connection string as a parameter
14. log the message of the connection string path to the session
15. use the following information to update information into the DVDT\_001\_DVD\_master in the database using SQL update make sure to assign the SQL update string to str SQL. The column name, size and datatype is information from the database. The value is the value the user gives. The where column indicates when to update the information

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Column name | Value | Datatype | Size | Where |
| DVD\_Title | @DVD\_Title | Char | 50 |  |
| DVD\_First | @DVD\_First | Char | 30 |  |
| DVD\_Last | @DVD\_Last | Char | 30 |  |
| DVD\_First2 | @DVD\_First2 | Char | 30 |  |
| DVD\_Last2 | @DVD\_Last2 | Char | 30 |  |
| DVD\_Company | @DVD\_Company | Char | 40 |  |
| DVD\_Genre | @DVD\_Genre | Small int |  |  |
| DVD\_Runtime | @DVD\_Runtime | Small int |  |  |
| DVD\_Type | @DVD\_Type | Char | 1 |  |
| DVD\_ID | @DVD\_ID | Big int |  | DVD\_ID =@DVD\_ID |

1. Declare a new database command and set it equal to a database command with with the SQL string and the database connection as parameters
2. do the next four steps for each parameter listed in the method. In the order which they are listed
3. Declare a new database parameter with the value, and datatype and size listed above as parameters
4. make the parameter direction input
5. make the parameter value equal to the parameter listed in the declaration of the method in the order in which they are listed
6. use database command parameters method to add the parameter
7. once the last parameter has been added open the connection to the database
8. execute the non-query command
9. close the connection to the database
10. **update actor1**
11. declare a string called strSQL set it equal to empty
12. declare a string called connstring set it equal to the correct string to connect the database
13. create a new database connection called cn set it equal to new database connection with the connection string as a parameter
14. log the message of the connection string path to the session
15. use the following information to update information into the DVDT\_001\_DVD\_master in the database using SQL update make sure to assign the SQL update string to str SQL. The column name, size and datatype is information from the database. The value is the value the user gives. The where column indicates when to update the information

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Column name | Value | Datatype | Size | Where |
| DVD\_First | @DVD\_First | Char | 30 |  |
| DVD\_Last | @DVD\_Last | Char | 30 |  |
| DVD\_ID | @DVD\_ID | Big int |  | DVD\_ID =@DVD\_ID |

1. Declare a new database command and set it equal to a database command with with the SQL string and the database connection as parameters
2. do the next four steps for each parameter listed in the method. In the order which they are listed
3. Declare a new database parameter with the value, and datatype and size listed above as parameters
4. make the parameter direction input
5. make the parameter value equal to the parameter listed in the declaration of the method in the order in which they are listed
6. use database command parameters method to add the parameter
7. once the last parameter has been added open the connection to the database
8. execute the non-query command
9. close the connection to the database
10. **update actor2**
11. declare a string called strSQL set it equal to empty
12. declare a string called connstring set it equal to the correct string to connect the database
13. create a new database connection called cn set it equal to new database connection with the connection string as a parameter
14. log the message of the connection string path to the session
15. use the following information to update information into the DVDT\_001\_DVD\_master in the database using SQL update make sure to assign the SQL update string to str SQL. The column name, size and datatype is information from the database. The value is the value the user gives. The where column indicates when to update the information

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Column name | Value | Datatype | Size | Where |
| DVD\_First2 | @DVD\_First2 | Char | 30 |  |
| DVD\_Last2 | @DVD\_Last2 | Char | 30 |  |
| DVD\_ID | @DVD\_ID | Big int |  | DVD\_ID =@DVD\_ID |

1. Declare a new database command and set it equal to a database command with with the SQL string and the database connection as parameters
2. do the next four steps for each parameter listed in the method. In the order which they are listed
3. Declare a new database parameter with the value, and datatype and size listed above as parameters
4. make the parameter direction input
5. make the parameter value equal to the parameter listed in the declaration of the method in the order in which they are listed
6. use database command parameters method to add the parameter
7. once the last parameter has been added open the connection to the database
8. execute the non-query command
9. close the connection to the database
10. **update company**
11. declare a string called strSQL set it equal to empty
12. declare a string called connstring set it equal to the correct string to connect the database
13. create a new database connection called cn set it equal to new database connection with the connection string as a parameter
14. log the message of the connection string path to the session
15. use the following information to update information into the DVDT\_001\_DVD\_master in the database using SQL update make sure to assign the SQL update string to str SQL. The column name, size and datatype is information from the database. The value is the value the user gives. The where column indicates when to update the information

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Column name | Value | Datatype | Size | Where |
| DVD\_Company | @DVD\_Company | Char | 40 |  |
| DVD\_ID | @DVD\_ID | Big int |  | DVD\_ID =@DVD\_ID |

1. Declare a new database command and set it equal to a database command with the SQL string and the database connection as parameters
2. do the next four steps for each parameter listed in the method. In the order which they are listed
3. Declare a new database parameter with the value, and datatype and size listed above as parameters
4. make the parameter direction input
5. make the parameter value equal to the parameter listed in the declaration of the method in the order in which they are listed
6. use database command parameters method to add the parameter
7. once the last parameter has been added open the connection to the database
8. execute the non-query command
9. close the connection to the database
10. **update genre**
11. declare a string called strSQL set it equal to empty
12. declare a string called connstring set it equal to the correct string to connect the database
13. create a new database connection called cn set it equal to new database connection with the connection string as a parameter
14. log the message of the connection string path to the session
15. use the following information to update information into the DVDT\_001\_DVD\_master in the database using SQL update make sure to assign the SQL update string to str SQL. The column name, size and datatype is information from the database. The value is the value the user gives. The where column indicates when to update the information

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Column name | Value | Datatype | Size | Where |
| DVD\_Genre | @DVD\_Genre | Small int |  |  |
| DVD\_ID | @DVD\_ID | Big int |  | DVD\_ID =@DVD\_ID |

1. Declare a new database command and set it equal to a database command with with the SQL string and the database connection as parameters
2. do the next four steps for each parameter listed in the method. In the order which they are listed
3. Declare a new database parameter with the value, and datatype and size listed above as parameters
4. make the parameter direction input
5. make the parameter value equal to the parameter listed in the declaration of the method in the order in which they are listed
6. use database command parameters method to add the parameter
7. once the last parameter has been added open the connection to the database
8. execute the non-query command
9. close the connection to the database
10. **update runtime**
11. declare a string called strSQL set it equal to empty
12. declare a string called connstring set it equal to the correct string to connect the database
13. create a new database connection called cn set it equal to new database connection with the connection string as a parameter
14. log the message of the connection string path to the session
15. use the following information to update information into the DVDT\_001\_DVD\_master in the database using SQL update make sure to assign the SQL update string to str SQL. The column name, size and datatype is information from the database. The value is the value the user gives. The where column indicates when to update the information

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Column name | Value | Datatype | Size | Where |
| DVD\_Runtime | @DVD\_Runtime | Small int |  |  |
| DVD\_ID | @DVD\_ID | Big int |  | DVD\_ID =@DVD\_ID |

1. Declare a new database command and set it equal to a database command with with the SQL string and the database connection as parameters
2. do the next four steps for each parameter listed in the method. In the order which they are listed
3. Declare a new database parameter with the value, and datatype and size listed above as parameters
4. make the parameter direction input
5. make the parameter value equal to the parameter listed in the declaration of the method in the order in which they are listed
6. use database command parameters method to add the parameter
7. once the last parameter has been added open the connection to the database
8. execute the non-query command
9. close the connection to the database
10. **update title**
11. declare a string called strSQL set it equal to empty
12. declare a string called connstring set it equal to the correct string to connect the database
13. create a new database connection called cn set it equal to new database connection with the connection string as a parameter
14. log the message of the connection string path to the session
15. use the following information to update information into the DVDT\_001\_DVD\_master in the database using SQL update make sure to assign the SQL update string to str SQL. The column name, size and datatype is information from the database. The value is the value the user gives. The where column indicates when to update the information

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Column name | Value | Datatype | Size | Where |
| DVD\_Title | @DVD\_Title | Char | 50 |  |
| DVD\_ID | @DVD\_ID | Big int |  | DVD\_ID =@DVD\_ID |

1. Declare a new database command and set it equal to a database command with with the SQL string and the database connection as parameters
2. do the next four steps for each parameter listed in the method. In the order which they are listed
3. Declare a new database parameter with the value, and datatype and size listed above as parameters
4. make the parameter direction input
5. make the parameter value equal to the parameter listed in the declaration of the method in the order in which they are listed
6. use database command parameters method to add the parameter
7. once the last parameter has been added open the connection to the database
8. execute the non-query command
9. close the connection to the database
10. **update type**
11. declare a string called strSQL set it equal to empty
12. declare a string called connstring set it equal to the correct string to connect the database
13. create a new database connection called cn set it equal to new database connection with the connection string as a parameter
14. log the message of the connection string path to the session
15. use the following information to update information into the DVDT\_001\_DVD\_master in the database using SQL update make sure to assign the SQL update string to str SQL. The column name, size and datatype is information from the database. The value is the value the user gives. The where column indicates when to update the information

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Column name | Value | Datatype | Size | Where |
| DVD\_Type | @DVD\_Type | Char | 1 |  |
| DVD\_ID | @DVD\_ID | Big int |  | DVD\_ID =@DVD\_ID |

1. Declare a new database command and set it equal to a database command with with the SQL string and the database connection as parameters
2. do the next four steps for each parameter listed in the method. In the order which they are listed
3. Declare a new database parameter with the value, and datatype and size listed above as parameters
4. make the parameter direction input
5. make the parameter value equal to the parameter listed in the declaration of the method in the order in which they are listed
6. use database command parameters method to add the parameter
7. once the last parameter has been added open the connection to the database
8. execute the non-query command
9. close the connection to the database
10. **select ID**
11. declare a string called strSQL set it equal to empty
12. declare a new database data reader and set it equal to null
13. declare a string called connstring set it equal to the correct string to connect the database
14. create a new database connection called cn set it equal to new database connection with the connection string as a parameter
15. log the message of the connection string path to the session
16. use the following information to select information from DVDT\_001\_DVD\_master in the database using SQL select make sure to assign the SQL select string to str SQL. The column name, and datatype is information from the database. The value is the value the user gives. The where column indicates when to select the information

|  |  |  |  |
| --- | --- | --- | --- |
| Column name | Value | Datatype | Where |
| DVD\_Title |  |  |  |
| DVD\_First |  |  |  |
| DVD\_Last |  |  |  |
| DVD\_First2 |  |  |  |
| DVD\_Last2 |  |  |  |
| DVD\_Company |  |  |  |
| DVD\_Genre |  |  |  |
| DVD\_Runtime |  |  |  |
| DVD\_Type |  |  |  |
| DVD\_Stat |  |  |  |
| DVD\_ID | @DVD\_ID | Big int | DVD\_ID =@DVD\_ID |

1. Declare a new database command and set it equal to a database command with with the SQL string and the database connection as parameters
2. do the next four steps for each parameter listed in the method. In the order which they are listed
3. Declare a new database parameter with the value, and datatype and size listed above as parameters
4. make the parameter direction input
5. make the parameter value equal to the parameter listed in the declaration of the method in the order in which they are listed
6. use database command parameters method to add the parameter
7. once the last parameter has been added open the connection to the database
8. execute the non-query command
9. set the database data reader object to the execute reader command. Make sure when the data reader is done reading the file it closes automatically
10. if the data reader is reading
11. set new ID equal to the value the data reader pulls out of the DVD\_ID column
12. set title equal to the value the data reader pulls out of the DVD\_Title column
13. set first equal to the value the data reader pulls out of the DVD\_First column
14. set last equal to the value the data reader pulls out of the DVD\_Last column
15. set first2 equal to the value the data reader pulls out of the DVD\_First2 column
16. set last2 equal to the value the data reader pulls out of the DVD\_Last2 column
17. set company to the value the data reader pulls out of the DVD\_Company column
18. set genre equal to the value the data reader pulls out of the DVD\_Genre column
19. set runtime equal to the value the data reader pulls out of the DVD\_Runtime column
20. set type equal to the value the data reader pulls out of the DVD\_Type column
21. set status equal to the value the data reader pulls out of the DVD\_Stat column
22. make sure to convert the values the data reader pulls out to the appropriate datatypes the private variables indicate
23. close if statement
24. else set the private variables equal to the following
25. newID = "0", title = "Title",first = "First", last = "Last", first2 = "First2", last2 = "Last2", company = "Company", genre = 0, runtime=0, type = 'U', status = 'E'
26. return data reader object
27. **select title**
28. declare a string called strSQL set it equal to empty
29. declare a new database data reader and set it equal to null
30. declare a string called connstring set it equal to the correct string to connect the database
31. create a new database connection called cn set it equal to new database connection with the connection string as a parameter
32. log the message of the connection string path to the session
33. use the following information to select information from DVDT\_001\_DVD\_master in the database using SQL select make sure to assign the SQL select string to str SQL. The column name, and datatype is information from the database. The value is the value the user gives. The where column indicates when to select the information

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Column name | Value | Datatype | Size | Where |
| DVD\_Title | @DVD\_Title | Char | 50 | DVD\_Title =@DVD\_Title |
| DVD\_First |  |  |  |  |
| DVD\_Last |  |  |  |  |
| DVD\_First2 |  |  |  |  |
| DVD\_Last2 |  |  |  |  |
| DVD\_Company |  |  |  |  |
| DVD\_Genre |  |  |  |  |
| DVD\_Runtime |  |  |  |  |
| DVD\_Type |  |  |  |  |
| DVD\_Stat |  |  |  |  |
| DVD\_ID |  |  |  |  |

1. Declare a new database command and set it equal to a database command with with the SQL string and the database connection as parameters
2. do the next four steps for each parameter listed in the method. In the order which they are listed
3. Declare a new database parameter with the value, and datatype and size listed above as parameters
4. make the parameter direction input
5. make the parameter value equal to the parameter listed in the declaration of the method in the order in which they are listed
6. use database command parameters method to add the parameter
7. once the last parameter has been added open the connection to the database
8. execute the non-query command
9. set the database data reader object to the execute reader command. Make sure when the data reader is done reading the file it closes automatically
10. if the data reader is reading
11. set new ID equal to the value the data reader pulls out of the DVD\_ID column
12. set title equal to the value the data reader pulls out of the DVD\_Title column
13. set first equal to the value the data reader pulls out of the DVD\_First column
14. set last equal to the value the data reader pulls out of the DVD\_Last column
15. set first2 equal to the value the data reader pulls out of the DVD\_First2 column
16. set last2 equal to the value the data reader pulls out of the DVD\_Last2 column
17. set company to the value the data reader pulls out of the DVD\_Company column
18. set genre equal to the value the data reader pulls out of the DVD\_Genre column
19. set runtime equal to the value the data reader pulls out of the DVD\_Runtime column
20. set type equal to the value the data reader pulls out of the DVD\_Type column
21. set status equal to the value the data reader pulls out of the DVD\_Stat column
22. make sure to convert the values the data reader pulls out to the appropriate datatypes the private variables indicate
23. close if statement
24. else set the private variables equal to the following
25. newID = "0", title = "Title",first = "First", last = "Last", first2 = "First2", last2 = "Last2", company = "Company", genre = 0, runtime=0, type = 'U', status = 'E'
26. return data reader object
27. **search actor1**
28. declare a string called strSQL set it equal to empty
29. declare a new database data reader and set it equal to null
30. declare link list of DVD objects called last name and set it equal to an empty link list
31. declare a string called connstring set it equal to the correct string to connect the database
32. create a new database connection called cn set it equal to new database connection with the connection string as a parameter
33. log the message of the connection string path to the session
34. use the following information to select information from DVDT\_001\_DVD\_master in the database using SQL select make sure to assign the SQL select string to str SQL. The column name, and datatype is information from the database. The value is the value the user gives. The where column indicates when to select the information

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Column name | Value | Datatype | Size | Where |
| DVD\_Title |  |  |  |  |
| DVD\_First |  |  |  |  |
| DVD\_Last | @DVD\_Last | Char | 30 | Where DVD\_Last like @DVD\_last |

1. Declare a new database command and set it equal to a database command with the SQL string and the database connection as parameters
2. do the next four steps for each parameter listed in the method. In the order which they are listed
3. Declare a new database parameter with the value, and datatype and size listed above as parameters
4. make the parameter direction input
5. make the parameter value equal to the parameter listed in the declaration of the method in the order in which they are listed make sure to append the % onto the parameter
6. use database command parameters method to add the parameter
7. once the last parameter has been added open the connection to the database
8. execute the non-query command
9. set the database data reader object to the execute reader command. Make sure when the data reader is done reading the file it closes automatically
10. while the data reader is reading
11. set title equal to the value the data reader pulls out of the DVD\_Title column
12. set first equal to the value the data reader pulls out of the DVD\_First column
13. set last equal to the value the data reader pulls out of the DVD\_Last column
14. make sure to convert the values the data reader pulls out to the appropriate datatypes the private variables indicate
15. declare a new DVD object and set it equal to the DVD constructor that has all the variables as parameters
16. add DVD object to last name as the last item
17. close while loop
18. return last name as an array
19. **search actor2**
20. declare a string called strSQL set it equal to empty
21. declare a new database data reader and set it equal to null
22. declare link list of DVD objects called last name2 and set it equal to an empty link list
23. declare a string called connstring set it equal to the correct string to connect the database
24. create a new database connection called cn set it equal to new database connection with the connection string as a parameter
25. log the message of the connection string path to the session
26. use the following information to select information from DVDT\_001\_DVD\_master in the database using SQL select make sure to assign the SQL select string to str SQL. The column name, and datatype is information from the database. The value is the value the user gives. The where column indicates when to select the information

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Column name | Value | Datatype | Size | Where |
| DVD\_Title |  |  |  |  |
| DVD\_First2 |  |  |  |  |
| DVD\_Last2 | @DVD\_Last2 | Char | 30 | Where DVD\_Last2 like @DVD\_Last2 |

1. Declare a new database command and set it equal to a database command with with the SQL string and the database connection as parameters
2. do the next four steps for each parameter listed in the method. In the order which they are listed
3. Declare a new database parameter with the value, and datatype and size listed above as parameters
4. make the parameter direction input
5. make the parameter value equal to the parameter listed in the declaration of the method in the order in which they are listed make sure to append the % onto the parameter
6. use database command parameters method to add the parameter
7. once the last parameter has been added open the connection to the database
8. execute the non-query command
9. set the database data reader object to the execute reader command. Make sure when the data reader is done reading the file it closes automatically
10. while the data reader is reading
11. set title equal to the value the data reader pulls out of the DVD\_Title column
12. set first2 equal to the value the data reader pulls out of the DVD\_First2 column
13. set last2 equal to the value the data reader pulls out of the DVD\_Last2 column
14. make sure to convert the values the data reader pulls out to the appropriate datatypes the private variables indicate
15. declare a new DVD object and set it equal to the DVD constructor that has all the variables as parameters
16. add DVD object to last name2 as the last item
17. close while loop
18. return last name2 as an array
19. **search company**
20. declare a string called strSQL set it equal to empty
21. declare a new database data reader and set it equal to null
22. declare link list of DVD objects called companies and set it equal to an empty link list
23. declare a string called connstring set it equal to the correct string to connect the database
24. create a new database connection called cn set it equal to new database connection with the connection string as a parameter
25. log the message of the connection string path to the session
26. use the following information to select information from DVDT\_001\_DVD\_master in the database using SQL select make sure to assign the SQL select string to str SQL. The column name, and datatype is information from the database. The value is the value the user gives. The where column indicates when to select the information

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Column name | Value | Datatype | Size | Where |
| DVD\_Title |  |  |  |  |
| DVD\_Company | @DVD\_Company | Char | 40 | Where DVD\_Company like @DVD\_Company |

1. Declare a new database command and set it equal to a database command with the SQL string and the database connection as parameters
2. do the next four steps for each parameter listed in the method. In the order which they are listed
3. Declare a new database parameter with the value, and datatype and size listed above as parameters
4. make the parameter direction input
5. make the parameter value equal to the parameter listed in the declaration of the method in the order in which they are listed make sure to append the % onto the parameter
6. use database command parameters method to add the parameter
7. once the last parameter has been added open the connection to the database
8. execute the non-query command
9. set the database data reader object to the execute reader command. Make sure when the data reader is done reading the file it closes automatically
10. while the data reader is reading
11. set title equal to the value the data reader pulls out of the DVD\_Title column
12. set company equal to the value the data reader pulls out of the DVD\_Company column
13. make sure to convert the values the data reader pulls out to the appropriate datatypes the private variables indicate
14. declare a new DVD object and set it equal to the DVD constructor that has all the variables as parameters
15. add DVD object to companies as the last item
16. close while loop
17. return companies as an array
18. **search title**
19. declare a string called strSQL set it equal to empty
20. declare a new database data reader and set it equal to null
21. declare link list of DVD objects called titles and set it equal to an empty link list
22. declare a string called connstring set it equal to the correct string to connect the database
23. create a new database connection called cn set it equal to new database connection with the connection string as a parameter
24. log the message of the connection string path to the session
25. use the following information to select information from DVDT\_001\_DVD\_master in the database using SQL select make sure to assign the SQL select string to str SQL. The column name, and datatype is information from the database. The value is the value the user gives. The where column indicates when to select the information

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Column name | Value | Datatype | Size | Where |
| DVD\_ID |  |  |  |  |
| DVD\_Title | @DVD\_Title | Char | 50 | Where DVD\_Title like @DVD\_Title |

1. Declare a new database command and set it equal to a database command with the SQL string and the database connection as parameters
2. do the next four steps for each parameter listed in the method. In the order which they are listed
3. Declare a new database parameter with the value, and datatype and size listed above as parameters
4. make the parameter direction input
5. make the parameter value equal to the parameter listed in the declaration of the method in the order in which they are listed make sure to append the % onto the parameter
6. use database command parameters method to add the parameter
7. once the last parameter has been added open the connection to the database
8. execute the non-query command
9. set the database data reader object to the execute reader command. Make sure when the data reader is done reading the file it closes automatically
10. while the data reader is reading
11. set title equal to the value the data reader pulls out of the DVD\_Title column
12. set new ID equal to the value the data reader pulls out of the DVD\_ID column
13. make sure to convert the values the data reader pulls out to the appropriate datatypes the private variables indicate
14. declare a new DVD object and set it equal to the DVD constructor that has all the variables as parameters
15. add DVD object to titles as the last item
16. close while loop
17. return titles as an array
18. **search status**
19. declare a string called strSQL set it equal to empty
20. declare a new database data reader and set it equal to null
21. declare link list of DVD objects called statuses and set it equal to an empty link list
22. declare a string called connstring set it equal to the correct string to connect the database
23. create a new database connection called cn set it equal to new database connection with the connection string as a parameter
24. log the message of the connection string path to the session
25. use the following information to select information from DVDT\_001\_DVD\_master in the database using SQL select make sure to assign the SQL select string to str SQL. The column name, and datatype is information from the database. The value is the value the user gives. The where column indicates when to select the information

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Column name | Value | Datatype | Size | Where |
| DVD\_ID |  |  |  |  |
| DVD\_Title |  |  |  |  |
| DVD\_Stat | @DVD\_Stat | Char | 1 | Where DVD\_Stat =@DVD\_Stat |

1. Declare a new database command and set it equal to a database command with the SQL string and the database connection as parameters
2. do the next four steps for each parameter listed in the method. In the order which they are listed
3. Declare a new database parameter with the value, and datatype and size listed above as parameters
4. make the parameter direction input
5. make the parameter value equal to the parameter listed in the declaration of the method in the order in which they are listed .
6. use database command parameters method to add the parameter
7. once the last parameter has been added open the connection to the database
8. execute the non-query command
9. set the database data reader object to the execute reader command. Make sure when the data reader is done reading the file it closes automatically
10. while the data reader is reading
11. set title equal to the value the data reader pulls out of the DVD\_Title column
12. set new ID equal to the value the data reader pulls out of the DVD\_ID column
13. set status equal to the value the data reader pulls out of the DVD\_Stat column
14. make sure to convert the values the data reader pulls out to the appropriate datatypes the private variables indicate
15. declare a new DVD object and set it equal to the DVD constructor that has all the variables as parameters
16. add DVD object to statuses as the last item
17. close while loop
18. return statuses as an array
19. **search genre**
20. declare a string called strSQL set it equal to empty
21. declare a new database data reader and set it equal to null
22. declare link list of DVD objects called codes and set it equal to an empty link list
23. declare a string called connstring set it equal to the correct string to connect the database
24. create a new database connection called cn set it equal to new database connection with the connection string as a parameter
25. log the message of the connection string path to the session
26. use the following information to select information from DVDT\_001\_DVD\_master in the database using SQL select make sure to assign the SQL select string to str SQL. The column name, and datatype is information from the database. The value is the value the user gives. The where column indicates when to select the information

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Column name | Value | Datatype | Size | Where |
| DVD\_Title |  |  |  |  |
| DVD\_Genre | @DVD\_Genre | Small int |  | Where DVD\_Genre =@DVD\_Genre |

1. Declare a new database command and set it equal to a database command with the SQL string and the database connection as parameters
2. do the next four steps for each parameter listed in the method. In the order which they are listed
3. Declare a new database parameter with the value, and datatype and size listed above as parameters
4. make the parameter direction input
5. make the parameter value equal to the parameter listed in the declaration of the method in the order in which they are listed .
6. use database command parameters method to add the parameter
7. once the last parameter has been added open the connection to the database
8. execute the non-query command
9. set the database data reader object to the execute reader command. Make sure when the data reader is done reading the file it closes automatically
10. while the data reader is reading
11. set title equal to the value the data reader pulls out of the DVD\_Title column
12. make sure to convert the values the data reader pulls out to the appropriate datatypes the private variables indicate
13. declare a new DVD object and set it equal to the DVD constructor that has all the variables as parameters
14. add DVD object to codes as the last item
15. close while loop
16. return codes as an array
17. **search runtime**
18. declare a string called strSQL set it equal to empty
19. declare a new database data reader and set it equal to null
20. declare link list of DVD objects called times and set it equal to an empty link list
21. declare a string called connstring set it equal to the correct string to connect the database
22. create a new database connection called cn set it equal to new database connection with the connection string as a parameter
23. log the message of the connection string path to the session
24. use the following information to select information from DVDT\_001\_DVD\_master in the database using SQL select make sure to assign the SQL select string to str SQL. The column name, and datatype is information from the database. The value is the value the user gives. The where column indicates when to select the information

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Column name | Value | Datatype | Size | Where |
| DVD\_Title |  |  |  |  |
| DVD\_Runtime | @DVD\_Runtime | Small int |  | Where DVD\_Runtime =@DVD\_Runtime |

1. Declare a new database command and set it equal to a database command with the SQL string and the database connection as parameters
2. do the next four steps for each parameter listed in the method. In the order which they are listed
3. Declare a new database parameter with the value, and datatype and size listed above as parameters
4. make the parameter direction input
5. make the parameter value equal to the parameter listed in the declaration of the method in the order in which they are listed .
6. use database command parameters method to add the parameter
7. once the last parameter has been added open the connection to the database
8. execute the non-query command
9. set the database data reader object to the execute reader command. Make sure when the data reader is done reading the file it closes automatically
10. while the data reader is reading
11. set title equal to the value the data reader pulls out of the DVD\_Title column
12. set runtime equal to the value the data reader pulls out of the DVD\_Runtime column
13. make sure to convert the values the data reader pulls out to the appropriate datatypes the private variables indicate
14. declare a new DVD object and set it equal to the DVD constructor that has all the variables as parameters
15. add DVD object to times as the last item
16. close while loop
17. return times as an array
18. **search type**
19. declare a string called strSQL set it equal to empty
20. declare a new database data reader and set it equal to null
21. declare link list of DVD objects called types and set it equal to an empty link list
22. declare a string called connstring set it equal to the correct string to connect the database
23. create a new database connection called cn set it equal to new database connection with the connection string as a parameter
24. log the message of the connection string path to the session
25. use the following information to select information from DVDT\_001\_DVD\_master in the database using SQL select make sure to assign the SQL select string to str SQL. The column name, and datatype is information from the database. The value is the value the user gives. The where column indicates when to select the information

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Column name | Value | Datatype | Size | Where |
| DVD\_Title |  |  |  |  |
| DVD\_Type | @DVD\_Type | Char | 1 | Where DVD\_Type =@DVD\_Type |

1. Declare a new database command and set it equal to a database command with the SQL string and the database connection as parameters
2. do the next four steps for each parameter listed in the method. In the order which they are listed
3. Declare a new database parameter with the value, and datatype and size listed above as parameters
4. make the parameter direction input
5. make the parameter value equal to the parameter listed in the declaration of the method in the order in which they are listed .
6. use database command parameters method to add the parameter
7. once the last parameter has been added open the connection to the database
8. execute the non-query command
9. set the database data reader object to the execute reader command. Make sure when the data reader is done reading the file it closes automatically
10. while the data reader is reading
11. set title equal to the value the data reader pulls out of the DVD\_Title column
12. set type equal to value the data reader pulls out of the DVD\_Type column
13. make sure to convert the values the data reader pulls out to the appropriate datatypes the private variables indicate
14. declare a new DVD object and set it equal to the DVD constructor that has all the variables as parameters
15. add DVD object to types as the last item
16. close while loop
17. return types as an array
18. **genre list**
19. declare a string called strSQL set it equal to empty
20. declare a new database data reader and set it equal to null
21. declare link list of Genre object called list and set it equal to an empty link list
22. declare a string called connstring set it equal to the correct string to connect the database
23. create a new database connection called cn set it equal to new database connection with the connection string as a parameter
24. log the message of the connection string path to the session
25. set the SQL string to select the genre and description columns from the DVDT\_001\_genre\_list
26. set the database data reader object to the execute reader command. Make sure when the data reader is done reading the file it closes automatically
27. while the data reader is reading
28. create new genre object using the constructor with both parameters.
29. Make sure to convert the values the data reader pulls out the columns to the appropriate datatypes
30. add the genre object as the last item to list
31. return list as an array

**class diagram**

|  |
| --- |
| **Session** |
| -Session ID: long  -static session\_id = 0: long  -DVD\_client: TCP client  -reader: stream reader  -writer: stream writer  -listening: Boolean  sa:serveractions |
| + Session (client: TCP client)-constructor  + static Logmessage (message: string, args: params object []): void  + Run (): void |

**Method descriptions**

1. Session-constructor

* sessionID = session\_id pre-incremented
* DVD\_client = client
* set reader to new stream reader with parameter client get stream
* set writer to new stream writer with parameter client get stream
* set the auto flush property of the writer equal to true
* listening = true
* log the following message to this session using the log message method. Client session session\_id initialized from the remote endpoint of the client converted to a string. The session ID and the remote endpoint values are the values stored in those variables

1. log message

* declare a string called s set it equal to string.format with message and args as parameters.
* Write the string along with the date and time separated by a tab to the screen
* if the server log writer equals null
* then create a new instance of log writer
* write the same message as was wrote to the screen with the same formats to the log

**Run Method Pseudocode**

1. **run**
2. declare a string called request and response and set them both equal to empty
3. while listening is true
4. open try
5. request equal the read line property of reader
6. catch I/O exception
7. log the following message client disconnected without shutting down the server first Session: session ID. Where session ID is the value of that variable
8. log the following message "request:", request Where request is the value of the variable
9. declaring new string called action and set it equal to the GetNodeText method in utilities class with Request/Action and request as parameters
10. if action not equal to null then
11. if action =addall then
12. open try
13. set response equal to the add all method from the server action class with request as a parameter
14. catch exception
15. set response equal to the add all method from the server action class with request and exception as parameters
16. close if statement
17. if action = addtitle then
18. open try
19. set response equal to the addtitle method from the server action class with request as a parameter
20. catch exception
21. set response equal to the addtitle method from the server action class with request and exception as parameters
22. close if statement
23. if action = updateall then
24. open try
25. set response equal to the update all method from the server action class with request as a parameter
26. catch exception
27. set response equal to the update all method from the server action class with request and exception as parameters
28. close if statement
29. if action = update title, then
30. open try
31. set response equal to the updatetitle method from the server action class with request as a parameter
32. catch exception
33. set response equal to the update title method from the server action class with request and exception as parameters
34. close if statement
35. if action = updateactor1then
36. open try
37. set response equal to the updateactor1method from the server action class with request as a parameter
38. catch exception
39. set response equal to the updateactor1method from the server action class with request and exception as parameters
40. close if statement
41. if action = updateactor2 then
42. open try
43. set response equal to the updateactor2 method from the server action class with request as a parameter
44. catch exception
45. set response equal to the updateactor2method from the server action class with request and exception as parameters
46. close if statement
47. if action = update company then
48. open try
49. set response equal to the update company method from the server action class with request as a parameter
50. catch exception
51. set response equal to the update company method from the server action class with request and exception as parameters
52. close if statement
53. if action = update genre then
54. open try
55. set response equal to the update genre method from the server action class with request as a parameter
56. catch exception
57. set response equal to the update genre method from the server action class with request and exception as parameters
58. close if statement
59. if action = update runtime then
60. open try
61. set response equal to the update runtime method from the server action class with request as a parameter
62. catch exception
63. set response equal to the update runtime method from the server action class with request and exception as parameters
64. close if statement
65. if action = update type then
66. open try
67. set response equal to the update type method from the server action class with request as a parameter
68. catch exception
69. set response equal to the update type method from the server action class with request and exception as parameters
70. close if statement
71. if action = search title then
72. open try
73. set response equal to the search title method from the server action class with request as a parameter
74. catch exception
75. set response equal to the search title method from the server action class with request and exception as parameters
76. close if statement
77. if action = search actor1then
78. open try
79. set response equal to the search actor1method from the server action class with request as a parameter
80. catch exception
81. set response equal to the search actor1 method from the server action class with request and exception as parameters
82. close if statement
83. if action = search actor2 then
84. open try
85. set response equal to the search actor2method from the server action class with request as a parameter
86. catch exception
87. set response equal to the search actor2 method from the server action class with request and exception as parameters
88. close if statement
89. if action = search company then
90. open try
91. set response equal to the search company method from the server action class with request as a parameter
92. catch exception
93. set response equal to the search company method from the server action class with request and exception as parameters
94. close if statement
95. if action = search genre then
96. open try
97. set response equal to the search genre method from the server action class with request as a parameter
98. catch exception
99. set response equal to the search genre method from the server action class with request and exception as parameters
100. close if statement
101. if action = search runtime then
102. open try
103. set response equal to the search runtime method from the server action class with request as a parameter
104. catch exception
105. set response equal to the search runtime method from the server action class with request and exception as parameters
106. close if statement
107. if action = search type then
108. open try
109. set response equal to the search type method from the server action class with request as a parameter
110. catch exception
111. set response equal to the search type method from the server action class with request and exception as parameters
112. close if statement
113. if action = search status then
114. open try
115. set response equal to the search status method from the server action class with request as a parameter
116. catch exception
117. set response equal to the search status method from the server action class with request and exception as parameters
118. close if statement
119. if action = select ID then
120. open try
121. set response equal to the select ID method from the server action class with request as a parameter
122. catch exception
123. set response equal to the select ID method from the server action class with request and exception as parameters
124. close if statement
125. if action = select title then
126. open try
127. set response equal to the select title method from the server action class with request as a parameter
128. catch exception
129. set response equal to the select title method from the server action class with request and exception as parameters
130. close if statement
131. if action = temp then
132. open try
133. set response equal to the delete from the server action class with request as a parameter
134. catch exception
135. set response equal to the delete method from the server action class with request and exception as parameters
136. close if statement
137. if action = purge then
138. open try
139. set response equal to the purge method from the server action class with request as a parameter
140. catch exception
141. set response equal to the purge method from the server action class with request and exception as parameters
142. close if statement
143. if action = recover then
144. open try
145. set response equal to the undo method from the server action class with request as a parameter
146. catch exception
147. set response equal to the undo method from the server action class with request and exception as parameters
148. close if statement
149. if action = get genre then
150. open try
151. set response equal to the get genres method from the server action class with request as a parameter
152. catch exception
153. set response equal to the get genres method from the server action class with request and exception as parameters
154. close if statement
155. if action = disconnect then
156. listening = false
157. close if statement
158. close if statement
159. log the response
160. write the response using the write line method of the writer
161. set both request and response to empty
162. close while loop
163. close reader, close writer, close client
164. log that the session number disconnected using session ID value

**class diagram**

|  |
| --- |
| **serveractions:genre** |
| -New ID: long  - genre ID: short  -returned ID: string  -DVD\_title: string  -DVD\_first: string  -DVD\_last: string  -DVD\_first2: string  -DVD\_last2: string  -DVD\_cp: string  -DVD\_genre: string  -DVD\_rt: string  -DVD\_type: string  -DVD\_stat: string  -description: string  -DVD\_ID: string  db: database  Genre list : genre [ ] |
| + Addall (request: string,ex=null: exception): string  + add title (request: string,ex=null: exception): string  + update all (request: string,ex=null: exception): string  + update title (request: string,ex=null: exception): string  + update actor 1(request: string,ex=null: exception): string  + update actor 2(request: string,ex=null: exception): string  + update company (request: string,ex=null: exception): string  + update genre (request: string,ex=null: exception): string  + update runtime (request: string,ex=null: exception): string  + update type (request: string,ex=null: exception): string  + delete (request: string,ex=null: exception): string  + undo (request: string,ex=null: exception): string  + purge (request: string,ex=null: exception): string  + select ID (request: string,ex=null: exception): string  + select title (request: string,ex=null: exception): string  + search title (request: string,ex=null: exception): string  + search actor 1(request: string,ex=null: exception): string  + search actor 2(request: string,ex=null: exception): string  + search company (request: string,ex=null: exception): string  + search genre (request: string,ex=null: exception): string  + search runtime (request: string,ex=null: exception): string  + search type (request: string,ex=null: exception): string  + search status (request: string,ex=null: exception): string |

**Serveractions methods pseudocode**

1. **AddAll**
2. declare a new string writer sw set it equal to an empty string writer
3. declare a new XML textwriter XML writer set it equal to a new XML textwriter with the string writer as a parameter
4. if ex is equals null then
5. DVD\_title = the get note text of the utilities method with Request/Title and request as parameters
6. DVD\_ first = the get note text of the utilities method with Request/First and request as parameters
7. DVD\_last = the get note text of the utilities method with Request/Last and request as parameters
8. DVD\_ first2 = the get note text of the utilities method with Request/First2 and request as parameters
9. DVD\_last2 = the get note text of the utilities method with Request/Last2 and request as parameters
10. DVD\_cp= the get note text of the utilities method with Request/Company and request as parameters
11. DVD\_genre = the get note text of the utilities method with Request/Genre and request as parameters
12. DVD\_rt= the get note text of the utilities method with Request/Runtime and request as parameters
13. DVD\_type = the get note text of the utilities method with Request/Type and request as parameters
14. DVD\_stat = the get note text of the utilities method with Request/Status and request as parameters
15. set new ID equal to the database add all method use the variables above as parameter values in the order which they are defined. Remember to convert when needed.
16. use the session log method to log the following message: record added! New ID: convert new ID into string
17. create an XML document with Response as the start element
18. Use ErrorCode and ID as tag values.
19. Make the error code value zero and make the ID value new ID
20. make sure to convert XML values to strings when needed
21. after the document has been created close XML writer
22. close if statement
23. open else statement
24. use the session log method to log the following message: Error: Error Occurred When Adding DVD
25. create an XML document with Response as the start element
26. Use ErrorCode and Message as tag values.
27. Make the error code value -1 and make the message value the following: Could not add DVD please try again
28. make sure to convert XML values to strings when needed
29. after the document has been created close XML writer
30. close else statement
31. return string writer as a string
32. **Add Title**
33. declare a new string writer sw set it equal to an empty string writer
34. declare a new XML textwriter XML writer set it equal to a new XML textwriter with the string writer as a parameter
35. if ex is equals null then
36. DVD\_title = the get note text of the utilities method with Request/Title and request as parameters
37. DVD\_stat = the get note text of the utilities method with Request/Status and request as parameters
38. set new ID equal to the database add title method use the variables above as parameter values in the order which they are defined. Remember to convert when necessary
39. use the session log method to log the following message: record added! New ID: convert new ID into string
40. create an XML document with Response as the start element
41. Use ErrorCode and ID as tag values.
42. Make the error code value zero and make the ID value new ID
43. make sure to convert XML values to strings when needed
44. after the document has been created close XML writer
45. close if statement
46. open else statement
47. use the session log method to log the following message: Error: Error Occurred When Adding DVD
48. create an XML document with Response as the start element
49. Use ErrorCode and Message as tag values.
50. Make the error code value -1 and make the message the following: Could not add DVD please try again
51. make sure to convert XML values to strings when needed
52. after the document has been created close XML writer
53. close else statement
54. return string writer as a string
55. **Delete**
56. declare a new string writer sw set it equal to an empty string writer
57. declare a new XML textwriter XML writer set it equal to a new XML textwriter with the string writer as a parameter
58. if ex is equals null then
59. DVD\_ID = the get note text of the utilities method with Request/ID and request as parameters
60. called the database delete method use the variables above as parameter values in the order which they are defined. Remember to convert when needed.
61. use the session log method to log the following message: record deleted! DVD ID: DVD ID variable deleted
62. create an XML document with Response as the start element
63. Use ErrorCode and Message as tag values.
64. Make the error code value zero and make message the following DVD ID: DVD ID variable deleted
65. make sure to convert XML values to strings when needed
66. after the document has been created close XML writer
67. close if statement
68. open else statement
69. use the session log method to log the following message: Error: Error Occurred When deleting DVD
70. create an XML document with Response as the start element
71. Use ErrorCode and Message as tag values.
72. Make the error code value -1 and make the message value the following: Could not delete DVD please try again
73. make sure to convert XML values to strings when needed
74. after the document has been created close XML writer
75. close else statement
76. return string writer as a string
77. **Purge**
78. declare a new string writer sw set it equal to an empty string writer
79. declare a new XML textwriter XML writer set it equal to a new XML textwriter with the string writer as a parameter
80. if ex is equals null then
81. DVD\_ID = the get note text of the utilities method with Request/ID and request as parameters
82. called the database purge method use the variables above as parameter values in the order which they are defined. Remember to convert when needed.
83. use the session log method to log the following message: record purged! DVD ID: DVD ID variable purged
84. create an XML document with Response as the start element
85. Use ErrorCode and Message as tag values.
86. Make the error code value zero and make message the following DVD ID: DVD ID variable purged
87. make sure to convert XML values to strings when needed
88. after the document has been created close XML writer
89. close if statement
90. open else statement
91. use the session log method to log the following message: Error: Error Occurred When purging DVD
92. create an XML document with Response as the start element
93. Use ErrorCode and Message as tag values.
94. Make the error code value -1 and make the message value the following: Could not purge DVD please try again
95. make sure to convert XML values to strings when needed
96. after the document has been created close XML writer
97. close else statement
98. return string writer as a string
99. **Undo**
100. declare a new string writer sw set it equal to an empty string writer
101. declare a new XML textwriter XML writer set it equal to a new XML textwriter with the string writer as a parameter
102. if ex is equals null then
103. DVD\_ID = the get note text of the utilities method with Request/ID and request as parameters
104. called the database undelete method use the variables above as parameter values in the order which they are defined. Remember to convert when needed.
105. use the session log method to log the following message: record recovered! DVD ID: DVD ID variable recovered
106. create an XML document with Response as the start element
107. Use ErrorCode and Message as tag values.
108. Make the error code value zero and make message the following DVD ID: DVD ID variable purged
109. make sure to convert XML values to strings when needed
110. after the document has been created close XML writer
111. close if statement
112. open else statement
113. use the session log method to log the following message: Error: Error Occurred When recovering DVD
114. create an XML document with Response as the start element
115. Use ErrorCode and Message as tag values.
116. Make the error code value -1 and make the message value the following: Could not recover DVD please try again
117. make sure to convert XML values to strings when needed
118. after the document has been created close XML writer
119. close else statement
120. return string writer as a string
121. **get genres**
122. declare a new string writer sw set it equal to an empty string writer
123. declare a new XML textwriter XML writer set it equal to a new XML textwriter with the string writer as a parameter
124. if ex is equals null then
125. set genre list equal to the database genre list method
126. use the session log method to log the following message: genres found
127. create an XML document with Response as the first start element
128. use ErrorCode as a tag, Make the error code value zero
129. Use Genres as a second starting tag
130. use a for each loop to loop through generalist use the genre object and use Genre as the variable
131. for each Genre in generalist use Genre as a third starting element
132. use ID and Description as tag
133. make sure to convert XML values to strings when needed
134. after the document has been created close XML writer
135. close if statement
136. open else statement
137. use the session log method to log the following message: genres not found
138. create an XML document with Response as the start element
139. Use ErrorCode and Message as tag values.
140. Make the error code value -1 and make the message value the following: Could not find the information please try again
141. make sure to convert XML values to strings when needed
142. after the document has been created close XML writer
143. close else statement
144. return string writer as a string
145. **search actor1**
146. declare a new string writer sw set it equal to an empty string writer
147. declare a new XML textwriter XML writer set it equal to a new XML textwriter with the string writer as a parameter
148. if ex is equals null then
149. DVD\_last = the get note text of the utilities method with Request/Last and request as parameters
150. create a last name array of DVD and set it equal to the database search actor1|method use the variables above as parameter values in the order which they are defined. Remember to convert when needed
151. use the session log method to log the following message: last names found
152. create an XML document with Response as the first start element
153. use ErrorCode as a tag, Make the error code value zero
154. Use DVDS as a second starting tag
155. use a for each loop to loop through lastnamelist use the DVD object and use movie as the variable
156. for each movie in last name last use DVD as a third starting element
157. use First, Last and Title as tag
158. use the appropriate getters to populate the XML document
159. make sure to convert XML values to strings when needed
160. after the document has been created close XML writer
161. close if statement
162. open else statement
163. use the session log method to log the following message: last names not found
164. create an XML document with Response as the start element
165. Use ErrorCode and Message as tag values.
166. Make the error code value -1 and make the message value the following: Could not find the information please try again
167. make sure to convert XML values to strings when needed
168. after the document has been created close XML writer
169. close else statement
170. return string writer as a string
171. **search actor2**
172. declare a new string writer sw set it equal to an empty string writer
173. declare a new XML textwriter XML writer set it equal to a new XML textwriter with the string writer as a parameter
174. if ex is equals null then
175. DVD\_last2 = the get note text of the utilities method with Request/Last2 and request as parameters
176. create a last name 2 list array of DVD and set it equal to the database search actor2|method use the variables above as parameter values in the order which they are defined. Remember to convert when needed
177. use the session log method to log the following message: last names found
178. create an XML document with Response as the first start element
179. use ErrorCode as a tag, Make the error code value zero
180. Use DVDS as a second starting tag
181. use a for each loop to loop through lastnamelist2 use the DVD object and use movie as the variable
182. for each movie in name last2 use DVD as a third starting element
183. use First2, Last2 and Title as tag
184. use the appropriate getters to populate the XML document
185. make sure to convert XML values to strings when needed
186. after the document has been created close XML writer
187. close if statement
188. open else statement
189. use the session log method to log the following message: last names not found
190. create an XML document with Response as the start element
191. Use ErrorCode and Message as tag values.
192. Make the error code value -1 and make the message value the following: Could not find the information please try again
193. make sure to convert XML values to strings when needed
194. after the document has been created close XML writer
195. close else statement
196. return string writer as a string
197. **search company**
198. declare a new string writer sw set it equal to an empty string writer
199. declare a new XML textwriter XML writer set it equal to a new XML textwriter with the string writer as a parameter
200. if ex is equals null then
201. DVD\_cp= the get note text of the utilities method with Request/Company and request as parameters
202. create a company list array of DVD and set it equal to the database search company |method use the variables above as parameter values in the order which they are defined. Remember to convert when needed
203. use the session log method to log the following message: companies found
204. create an XML document with Response as the first start element
205. use ErrorCode as a tag, Make the error code value zero
206. Use DVDS as a second starting tag
207. use a for each loop to loop through companies list use the DVD object and use movie as the variable
208. for each movie in companies list use DVD as a third starting element
209. use Company and Title as tag
210. use the appropriate getters to populate the XML document
211. make sure to convert XML values to strings when needed
212. after the document has been created close XML writer
213. close if statement
214. open else statement
215. use the session log method to log the following message: companies not found
216. create an XML document with Response as the start element
217. Use ErrorCode and Message as tag values.
218. Make the error code value -1 and make the message value the following: Could not find the information please try again
219. make sure to convert XML values to strings when needed
220. after the document has been created close XML writer
221. close else statement
222. return string writer as a string
223. **search genre**
224. declare a new string writer sw set it equal to an empty string writer
225. declare a new XML textwriter XML writer set it equal to a new XML textwriter with the string writer as a parameter
226. if ex is equals null then
227. set genre list equal to the database generalist method
228. DVD\_genre = the get note text of the utilities method with Request/Genre and request as parameters
229. create a descriptions array of DVD and set it equal to the database search genre method use the variables above as parameter values in the order which they are defined. Remember to convert when needed
230. set genre ID equal to DVD\_genre converted to a short
231. use a for each loop to loop through generalist use the genre object and use movie genre as the variable
232. if the genre ID from the movie genre equals the genre ID from step seven then
233. description = movie genre get description method
234. set the genre description equal to description
235. close for loop
236. use the session log method to log the following message: movie genres found
237. create an XML document with Response as the first start element
238. use ErrorCode as a tag, Make the error code value zero
239. Use DVDS as a second starting tag
240. use a for each loop to loop through descriptions use the DVD object and use movie as the variable
241. for each movie in description use DVD as a third starting element
242. use Description and Title as tag
243. use the appropriate getters to populate the XML document
244. make sure to convert XML values to strings when needed
245. after the document has been created close XML writer
246. close if statement
247. open else statement
248. use the session log method to log the following message: movie genres not found
249. create an XML document with Response as the start element
250. Use ErrorCode and Message as tag values.
251. Make the error code value -1 and make the message value the following: Could not find the information please try again
252. make sure to convert XML values to strings when needed
253. after the document has been created close XML writer
254. close else statement
255. return string writer as a string
256. **search runtime**
257. declare a new string writer sw set it equal to an empty string writer
258. declare a new XML textwriter XML writer set it equal to a new XML textwriter with the string writer as a parameter
259. if ex is equals null then
260. DVD\_rt = the get note text of the utilities method with Request/Runtime and request as parameters
261. create a time list array of DVD and set it equal to the database search runtime method use the variables above as parameter values in the order which they are defined. Remember to convert when needed
262. use the session log method to log the following message: runtimes found
263. create an XML document with Response as the first start element
264. use ErrorCode as a tag, Make the error code value zero
265. Use DVDS as a second starting tag
266. use a for each loop to loop through times list use the DVD object and use movie as the variable
267. for each movie in time list use DVD as a third starting element
268. use Runtime and Title as tag
269. use the appropriate getters to populate the XML document
270. make sure to convert XML values to strings when needed
271. after the document has been created close XML writer
272. close if statement
273. open else statement
274. use the session log method to log the following message: last names not found
275. create an XML document with Response as the start element
276. Use ErrorCode and Message as tag values.
277. Make the error code value -1 and make the message value the following: Could not find the information please try again
278. make sure to convert XML values to strings when needed
279. after the document has been created close XML writer
280. close else statement
281. return string writer as a string
282. **search status**
283. declare a new string writer sw set it equal to an empty string writer
284. declare a new XML textwriter XML writer set it equal to a new XML textwriter with the string writer as a parameter
285. if ex is equals null then
286. DVD\_stat = the get note text of the utilities method with Request/Status and request as parameters
287. create a status list array of DVD and set it equal to the database search status method use the variables above as parameter values in the order which they are defined. Remember to convert when needed
288. use the session log method to log the following message: DVD statuses found
289. create an XML document with Response as the first start element
290. use ErrorCode as a tag, Make the error code value zero
291. Use DVDS as a second starting tag
292. use a for each loop to loop through status list use the DVD object and use movie as the variable
293. for each movie in status list use DVD as a third starting element
294. use Status, ID and Title as tag
295. use the appropriate getters to populate the XML document
296. make sure to convert XML values to strings when needed
297. after the document has been created close XML writer
298. close if statement
299. open else statement
300. use the session log method to log the following message: DVD statuses not found
301. create an XML document with Response as the start element
302. Use ErrorCode and Message as tag values.
303. Make the error code value -1 and make the message value the following: Could not find the information please try again
304. make sure to convert XML values to strings when needed
305. after the document has been created close XML writer
306. close else statement
307. return string writer as a string
308. **search title**
309. declare a new string writer sw set it equal to an empty string writer
310. declare a new XML textwriter XML writer set it equal to a new XML textwriter with the string writer as a parameter
311. if ex is equals null then
312. DVD\_title = the get note text of the utilities method with Request/Title and request as parameters
313. create a title list array of DVD and set it equal to the database search title method use the variables above as parameter values in the order which they are defined. Remember to convert when needed
314. use the session log method to log the following message: titles found
315. create an XML document with Response as the first start element
316. use ErrorCode as a tag, Make the error code value zero
317. Use DVDS as a second starting tag
318. use a for each loop to loop through title list use the DVD object and use movie as the variable
319. for each movie in title list use DVD as a third starting element
320. use ID and Title as tag
321. use the appropriate getters to populate the XML document
322. make sure to convert XML values to strings when needed
323. after the document has been created close XML writer
324. close if statement
325. open else statement
326. use the session log method to log the following message: titles not found
327. create an XML document with Response as the start element
328. Use ErrorCode and Message as tag values.
329. Make the error code value -1 and make the message value the following: Could not find the information please try again
330. make sure to convert XML values to strings when needed
331. after the document has been created close XML writer
332. close else statement
333. return string writer as a string
334. **search type**
335. declare a new string writer sw set it equal to an empty string writer
336. declare a new XML textwriter XML writer set it equal to a new XML textwriter with the string writer as a parameter
337. if ex is equals null then
338. DVD\_type = the get note text of the utilities method with Request/Type and request as parameters
339. create a types list array of DVD and set it equal to the database search type method use the variables above as parameter values in the order which they are defined. Remember to convert when needed
340. use the session log method to log the following message: DVD types found
341. create an XML document with Response as the first start element
342. use ErrorCode as a tag, Make the error code value zero
343. Use DVDS as a second starting tag
344. use a for each loop to loop through types list use the DVD object and use movie as the variable
345. for each movie in types list use DVD as a third starting element
346. use Type and Title as tag
347. make sure to convert XML values to strings when needed
348. use the appropriate getters to populate the XML document
349. after the document has been created close XML writer
350. close if statement
351. open else statement
352. use the session log method to log the following message: types not found
353. create an XML document with Response as the start element
354. Use ErrorCode and Message as tag values.
355. Make the error code value -1 and make the message value the following: Could not find the information please try again
356. make sure to convert XML values to strings when needed
357. after the document has been created close XML writer
358. close else statement
359. return string writer as a string
360. **Update All**
361. declare a new string writer sw set it equal to an empty string writer
362. declare a new XML textwriter XML writer set it equal to a new XML textwriter with the string writer as a parameter
363. if ex is equals null then
364. DVD\_ID = the get note text of the utilities method with Request/ID and request as parameters
365. DVD\_title = the get note text of the utilities method with Request/Title and request as parameters
366. DVD\_ first = the get note text of the utilities method with Request/First and request as parameters
367. DVD\_last = the get note text of the utilities method with Request/Last and request as parameters
368. DVD\_ first2 = the get note text of the utilities method with Request/First2 and request as parameters
369. DVD\_last2 = the get note text of the utilities method with Request/Last2 and request as parameters
370. DVD\_cp= the get note text of the utilities method with Request/Company and request as parameters
371. DVD\_genre = the get note text of the utilities method with Request/Genre and request as parameters
372. DVD\_rt= the get note text of the utilities method with Request/Runtime and request as parameters
373. DVD\_type = the get note text of the utilities method with Request/Type and request as parameters
374. DVD\_stat = the get note text of the utilities method with Request/Status and request as parameters
375. call the database update all method use the variables above as parameter values in the order which they are defined. Remember to convert when needed.
376. use the session log method to log the following message: record updated DVD ID: DVD ID variable value updated
377. create an XML document with Response as the start element
378. Use ErrorCode and Message as tag values.
379. Make the error code value zero and make the Message: DVD ID: DVD ID variable value updated
380. make sure to convert XML values to strings when needed
381. after the document has been created close XML writer
382. close if statement
383. open else statement
384. use the session log method to log the following message: Error: Error Occurred when updating
385. create an XML document with Response as the start element
386. Use ErrorCode and Message as tag values.
387. Make the error code value -1 and make the message value the following: Could not update DVD please try again
388. make sure to convert XML values to strings when needed
389. after the document has been created close XML writer
390. close else statement
391. return string writer as a string
392. **Update actor1**
393. declare a new string writer sw set it equal to an empty string writer
394. declare a new XML textwriter XML writer set it equal to a new XML textwriter with the string writer as a parameter
395. if ex is equals null then
396. DVD\_ID = the get note text of the utilities method with Request/ID and request as parameters
397. DVD\_ first = the get note text of the utilities method with Request/First and request as parameters
398. DVD\_last = the get note text of the utilities method with Request/Last and request as parameters
399. call the database update actor 1method use the variables above as parameter values in the order which they are defined. Remember to convert when needed.
400. use the session log method to log the following message: record updated DVD ID: DVD ID variable value updated
401. create an XML document with Response as the start element
402. Use ErrorCode and Message as tag values.
403. Make the error code value zero and make the Message: DVD ID: DVD ID variable value Actor 1 updated
404. make sure to convert XML values to strings when needed
405. after the document has been created close XML writer
406. close if statement
407. open else statement
408. use the session log method to log the following message: Error: Error Occurred when updating
409. create an XML document with Response as the start element
410. Use ErrorCode and Message as tag values.
411. Make the error code value -1 and make the message value the following: Could not update DVD Actor 1 updated please try again
412. make sure to convert XML values to strings when needed
413. after the document has been created close XML writer
414. close else statement
415. return string writer as a string
416. **update actor2**
417. declare a new string writer sw set it equal to an empty string writer
418. declare a new XML textwriter XML writer set it equal to a new XML textwriter with the string writer as a parameter
419. if ex is equals null then
420. DVD\_ID = the get note text of the utilities method with Request/ID and request as parameters
421. DVD\_ first2 = the get note text of the utilities method with Request/First2 and request as parameters
422. DVD\_last2 = the get note text of the utilities method with Request/Last2 and request as parameters
423. call the database update actor 2 method use the variables above as parameter values in the order which they are defined. Remember to convert when needed.
424. use the session log method to log the following message: record updated DVD ID: DVD ID variable value updated
425. create an XML document with Response as the start element
426. Use ErrorCode and Message as tag values.
427. Make the error code value zero and make the Message: DVD ID: DVD ID variable value Actor 2 updated
428. make sure to convert XML values to strings when needed
429. after the document has been created close XML writer
430. close if statement
431. open else statement
432. use the session log method to log the following message: Error: Error Occurred when updating
433. create an XML document with Response as the start element
434. Use ErrorCode and Message as tag values.
435. Make the error code value -1 and make the message value the following: Could not update DVD Actor 2 please try again
436. make sure to convert XML values to strings when needed
437. after the document has been created close XML writer
438. close else statement
439. return string writer as a string
440. **Update company**
441. declare a new string writer sw set it equal to an empty string writer
442. declare a new XML textwriter XML writer set it equal to a new XML textwriter with the string writer as a parameter
443. if ex is equals null then
444. DVD\_ID = the get note text of the utilities method with Request/ID and request as parameters
445. DVD\_cp= the get note text of the utilities method with Request/Company and request as parameters
446. call the database update company method use the variables above as parameter values in the order which they are defined. Remember to convert when needed.
447. use the session log method to log the following message: record updated DVD ID: DVD ID variable value updated
448. create an XML document with Response as the start element
449. Use ErrorCode and Message as tag values.
450. Make the error code value zero and make the Message: DVD ID: DVD ID variable value production company updated
451. make sure to convert XML values to strings when needed
452. after the document has been created close XML writer
453. close if statement
454. open else statement
455. use the session log method to log the following message: Error: Error Occurred when updating
456. create an XML document with Response as the start element
457. Use ErrorCode and Message as tag values.
458. Make the error code value -1 and make the message value the following: Could not update DVD production company please try again
459. make sure to convert XML values to strings when needed
460. after the document has been created close XML writer
461. close else statement
462. return string writer as a string
463. **Update genre**
464. declare a new string writer sw set it equal to an empty string writer
465. declare a new XML textwriter XML writer set it equal to a new XML textwriter with the string writer as a parameter
466. if ex is equals null then
467. DVD\_ID = the get note text of the utilities method with Request/ID and request as parameters
468. DVD\_genre = the get note text of the utilities method with Request/Genre and request as parameters
469. call the database update genre method use the variables above as parameter values in the order which they are defined. Remember to convert when needed.
470. use the session log method to log the following message: record updated DVD ID: DVD ID variable value updated
471. create an XML document with Response as the start element
472. Use ErrorCode and Message as tag values.
473. Make the error code value zero and make the Message: DVD ID: DVD ID variable value genre code updated
474. make sure to convert XML values to strings when needed
475. after the document has been created close XML writer
476. close if statement
477. open else statement
478. use the session log method to log the following message: Error: Error Occurred when updating
479. create an XML document with Response as the start element
480. Use ErrorCode and Message as tag values.
481. Make the error code value -1 and make the message value the following: Could not update DVD genre code please try again
482. make sure to convert XML values to strings when needed
483. after the document has been created close XML writer
484. close else statement
485. return string writer as a string
486. **update runtime**
487. declare a new string writer sw set it equal to an empty string writer
488. declare a new XML textwriter XML writer set it equal to a new XML textwriter with the string writer as a parameter
489. if ex is equals null then
490. DVD\_ID = the get note text of the utilities method with Request/ID and request as parameters
491. DVD\_rt= the get note text of the utilities method with Request/Runtime and request as parameters
492. call the database update runtime method use the variables above as parameter values in the order which they are defined. Remember to convert when needed.
493. use the session log method to log the following message: record updated DVD ID: DVD ID variable value updated
494. create an XML document with Response as the start element
495. Use ErrorCode and Message as tag values.
496. Make the error code value zero and make the Message: DVD ID: DVD ID variable value runtime updated
497. make sure to convert XML values to strings when needed
498. after the document has been created close XML writer
499. close if statement
500. open else statement
501. use the session log method to log the following message: Error: Error Occurred when updating
502. create an XML document with Response as the start element
503. Use ErrorCode and Message as tag values.
504. Make the error code value -1 and make the message value the following: Could not update DVD runtime please try again
505. make sure to convert XML values to strings when needed
506. after the document has been created close XML writer
507. close else statement
508. return string writer as a string
509. **Update title**
510. declare a new string writer sw set it equal to an empty string writer
511. declare a new XML textwriter XML writer set it equal to a new XML textwriter with the string writer as a parameter
512. if ex is equals null then
513. DVD\_ID = the get note text of the utilities method with Request/ID and request as parameters
514. DVD\_title = the get note text of the utilities method with Request/Title and request as parameters
515. call the database update title method use the variables above as parameter values in the order which they are defined. Remember to convert when needed.
516. use the session log method to log the following message: record updated DVD ID: DVD ID variable value updated
517. create an XML document with Response as the start element
518. Use ErrorCode and Message as tag values.
519. Make the error code value zero and make the Message: DVD ID: DVD ID variable value title updated
520. make sure to convert XML values to strings when needed
521. after the document has been created close XML writer
522. close if statement
523. open else statement
524. use the session log method to log the following message: Error: Error Occurred when updating
525. create an XML document with Response as the start element
526. Use ErrorCode and Message as tag values.
527. Make the error code value -1 and make the message value the following: Could not update DVD title please try again
528. make sure to convert XML values to strings when needed
529. after the document has been created close XML writer
530. close else statement
531. return string writer as a string
532. **Update type**
533. declare a new string writer sw set it equal to an empty string writer
534. declare a new XML textwriter XML writer set it equal to a new XML textwriter with the string writer as a parameter
535. if ex is equals null then
536. DVD\_ID = the get note text of the utilities method with Request/ID and request as parameters
537. DVD\_type = the get note text of the utilities method with Request/Type and request as parameters
538. call the database update type method use the variables above as parameter values in the order which they are defined. Remember to convert when needed.
539. use the session log method to log the following message: record updated DVD ID: DVD ID variable value updated
540. create an XML document with Response as the start element
541. Use ErrorCode and Message as tag values.
542. Make the error code value zero and make the Message: DVD ID: DVD ID variable value type updated
543. make sure to convert XML values to strings when needed
544. after the document has been created close XML writer
545. close if statement
546. open else statement
547. use the session log method to log the following message: Error: Error Occurred when updating
548. create an XML document with Response as the start element
549. Use ErrorCode and Message as tag values.
550. Make the error code value -1 and make the message value the following: Could not update DVD type please try again
551. make sure to convert XML values to strings when needed
552. after the document has been created close XML writer
553. close else statement
554. return string writer as a string
555. **select ID**
556. declare a new string writer sw set it equal to an empty string writer
557. declare a new XML textwriter XML writer set it equal to a new XML textwriter with the string writer as a parameter
558. if ex is equals null then
559. set genre list equal to the database generalist method
560. DVD\_ID = the get note text of the utilities method with Request/ID and request as parameters
561. create a data reader object called dr said it equal to the database select ID method use the variables above as parameter values in the order which they are defined. Remember to convert when needed
562. set returned ID equal to DVD\_ID value in the data reader converted to a string
563. set DVD\_title equal to DVD\_Title value in the data reader converted to a string
564. set DVD\_first equal to DVD\_First value in the data reader converted to a string
565. set DVD\_last equal to DVD\_Last value in the data reader converted to a string
566. set DVD\_firs2 equal to DVD\_First2 value in the data reader converted to a string
567. set DVD\_last2 equal to DVD\_Last2 value in the data reader converted to a string
568. set DVD\_cp equal to DVD\_Company value in the data reader converted to a string
569. set DVD\_rt equal to DVD\_Runtime value in the data reader converted to a string
570. set DVD\_type equal to DVD\_Type value in the data reader converted to a string
571. set DVD\_stat equal to DVD\_Stat value in the data reader converted to a string
572. set genreID equal to DVD\_Genre value in the data reader converted to a short
573. use a for each loop to loop through generalist use the genre object and use movie genre as the variable
574. if the genre ID from the movie genre equals the genre ID from step seven then
575. description = movie genre get description method
576. set the genre description equal to description
577. close for loop
578. use the session log method to log the following message: record found
579. create an XML document with Response as the first start element
580. Use ErrorCode, ID, Title, First, Last, First2, Last2 , Company, Genre, Description, Runtime, Type, and Status as tag values.
581. Make the error code value zero, make the ID the value of returned ID, make the Title of the value of DVD\_title, make the First of the value of DVD\_first, make the Last of the value of DVD\_ last, make the First2of the value of DVD\_first2, make the Last2 of the value of DVD\_ last2, make the Company of the value of DVD\_cp, make the Genre of the value of genre ID make the Description of the value of the description getter, make the Runtime of the value of DVD\_rt , make the Type of the value of DVD\_type, make the Status of the value of DVD\_stat.
582. make sure to convert XML values to strings when needed
583. after the document has been created close XML writer
584. close if statement
585. open else statement
586. use the session log method to log the following message: no record found
587. create an XML document with Response as the start element
588. Use ErrorCode and Message as tag values.
589. Make the error code value -1 and make the message value the following: Could not find the information please try again
590. make sure to convert XML values to strings when needed
591. after the document has been created close XML writer
592. close else statement
593. return string writer as a string
594. **select title**
595. declare a new string writer sw set it equal to an empty string writer
596. declare a new XML textwriter XML writer set it equal to a new XML textwriter with the string writer as a parameter
597. if ex is equals null then
598. set genre list equal to the database generalist method
599. DVD\_title = the get note text of the utilities method with Request/Title and request as parameters
600. create a data reader object called dr said it equal to the database select title method use the variables above as parameter values in the order which they are defined. Remember to convert when needed
601. set returned ID equal to DVD\_ID value in the data reader converted to a string
602. set DVD\_title equal to DVD\_Title value in the data reader converted to a string
603. set DVD\_first equal to DVD\_First value in the data reader converted to a string
604. set DVD\_last equal to DVD\_Last value in the data reader converted to a string
605. set DVD\_firs2 equal to DVD\_First2 value in the data reader converted to a string
606. set DVD\_last2 equal to DVD\_Last2 value in the data reader converted to a string
607. set DVD\_cp equal to DVD\_Company value in the data reader converted to a string
608. set DVD\_rt equal to DVD\_Runtime value in the data reader converted to a string
609. set DVD\_type equal to DVD\_Type value in the data reader converted to a string
610. set DVD\_stat equal to DVD\_Stat value in the data reader converted to a string
611. set genreID equal to DVD\_Genre value in the data reader converted to a short
612. use a for each loop to loop through generalist use the genre object and use movie genre as the variable
613. if the genre ID from the movie genre equals the genre ID from step seven then
614. description = movie genre get description method
615. set the genre description equal to description
616. close for loop
617. use the session log method to log the following message: record found
618. create an XML document with Response as the first start element
619. Use ErrorCode, ID, Title, First, Last, First2, Last2 , Company, Genre, Description, Runtime, Type, and Status as tag values.
620. Make the error code value zero, make the ID the value of returned ID, make the Title of the value of DVD\_title, make the First of the value of DVD\_first, make the Last of the value of DVD\_ last, make the First2of the value of DVD\_first2, make the Last2 of the value of DVD\_ last2, make the Company of the value of DVD\_cp, make the Genre of the value of genre ID make the Description of the value of the description getter, make the Runtime of the value of DVD\_rt , make the Type of the value of DVD\_type, make the Status of the value of DVD\_stat.
621. make sure to convert XML values to strings when needed
622. after the document has been created close XML writer
623. close if statement
624. open else statement
625. use the session log method to log the following message: no record found
626. create an XML document with Response as the start element
627. Use ErrorCode and Message as tag values.
628. Make the error code value -1 and make the message value the following: Could not find the information please try again
629. make sure to convert XML values to strings when needed
630. after the document has been created close XML writer
631. close else statement
632. return string writer as a string

**class diagram**

|  |
| --- |
| **server** |
| -Static M\_port = 5000: int  -static listening = true: Boolean  -static DVD\_client =null: TCP client  -static listener =null: TCP listener  + static log writer = null: log writer |
| - static Logmessage (message: string, args: params object []): void  static Main (args: string []): void |

**Method Descriptions**

1. log message

* declare a string called s set it equal to string.format with message and args as parameters.
* Write the string along with the date and time separated by a tab to the screen
* if the server log writer equals null
* then create a new instance of log writer
* write the same message as was wrote to the screen with the same formats to the log

**Main Method Pseudocode**

1. **Main**
2. log and display the following message using the server log message method. Start system initialization.
3. Set listener to a new TCP listener with system.net.IP address.any and M\_port as parameters.
4. Start the listener
5. login display the following message. Server started
6. while listening
7. login display the following message. Awaiting connection on port M\_port. Where M\_port is the value of that variable
8. DVD\_client = the accept TCP client method of the listener
9. login display the following message accepted connection from the remote endpoint method of the client converted to a string. Where the remote endpoint is the local host address of the PC
10. create a new session and set it equal to a new session using DVD\_client as a parameter
11. declare thread called new thread
12. set the new thread object equal to new thread with the session run method as a parameter.
13. Start new thread
14. close while loop
15. log and display the following message: server shutdown initialized
16. stop listener
17. login display the following message: server shutdown completed