**Software Testing**

**Assignment#2**

**Nada Mohamed Fathy Omar** [20186031](mailto:s.makady@fci-cu.edu.eg)

**Ayat Hany Ahmed**  20186007

Group : S1

**Class#1**

**(ApplicationService)**

* **Method1(getCurrentWeather(String cityName)):**
* **Partitions**:

**\*For string cityName we have 4 partitions:**

* B1:empty string
* B2:null string
* B3:wrong city name
* B4:non empty string

\***For output :**

* **C1**: "invalid input"
* **C2**: "invalid response"
* **C3:** Actual response
* **Boundaries**:

\***For string cityName we have**

* There are no upper bounds for s and txt (i.e., no upper limit on their length).(valid)
* The lower bound on their length is 0, or null .(invalid)
* **The chosen coverage criteria with complete explanation:**
* we used ACOC because we had used all the combination of partitions from all characteristics, that results 4 test Cases :
* t1: **cityName** =” “ (covers B1,C1)
* t2: **cityName =**null (covers B2,C1)
* t3: **cityName =** wrong city name”lodon” (covers B3,C2)
* t4: **cityName** = non empty string “London”(covers B4,C3)
* **Method2(getCurrentWeather(int cityId)):**
* **Partitions**:

**\*For int cityId we have 4 partitions:**

* B1:negative(<0)
* B2:equals 0
* B3:positive(>0)(id with no city)
* B4:positive(>0)(id with city)

\***For output :**

* **C1**: "invalid input"
* **C2**: "invalid response"
* **C3:** Actual response
* **Boundaries**:

\***For integer CityId**

* valid for n>0, invalid for n<=0
* **The chosen coverage criteria with complete explanation:**
* we used ACOC because we had used all the combination of partitions from all characteristics, that results 4 test Cases :
* t1: **cityId** =negative(-2) (covers B1,C1)
* t2: **cityId =**0(covers B2,C1)
* t3: **cityId =**800(covers B3,C2)
* t4: **cityId =** positive(2643743) (covers B4,C3)
* **Method3(getCurrentWeather(double latitude, double longitude)):**
* **Partitions**:

**\*For double latitude we have 3 partitions:**

* A1:negative(<0)
* A2:equals 0
* A3:positive(>0)

**\*For double longitude we have 3 partitions:**

* B1:negative(<0)
* B2:equals 0
* B3:positive(>0)

\***For output :**

* **C1**: "invalid input"
* **C2**: "invalid response"
* **C3:** Actual response
* **Boundaries**:

\***For double** latitude/ longitude

* valid for latitude >/</= 0 ,longitude >/</=0, invalid for latitude =0 && longitude=0
* **The chosen coverage criteria with complete explanation:**
* We used BCC because we choose to use the base choice for each characteristic, that results 5 testCases:
* t1: latitude =negative(-0.1257), longitude =positive (51.5085)(base case/happy path) (covers A1 ,B3,C3)
* t2: latitude =negative(-1), longitude =negative (-1) (covers A1 ,B1,C2)
* t3: latitude =negative(-0.1257), longitude =zero (0) (covers A1 ,B2,C3)
* t4: latitude =zero(0), longitude =zero (0) (covers A1 ,B2,C1)
* t5: latitude =positive(0.1257), longitude =zero (0) (covers A3 ,B2,C3)
* **Report for all testcases**:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Method** | **Params** | **Returns** | **Values** | **Ch ID** | **Character -istic** |
| **getCurrentWeather** | String  CityName | string | Response with the weather details of mentioned city name as string, or “invalid input", or "invalid response”. | C1 | Response with the weather details of mentioned city name as string, or error message consider to the situation. |
| **getCurrentWeather** | String  CityId | string | Response with the weather details of mentioned city name as string, or “invalid input", or "invalid response”. | C2 | Response with the weather details of mentioned city name as string, or error message consider to the situation. |
| **getCurrentWeather** | double latitude, double longitude | string | Response with the weather details of mentioned city name as string, or “invalid input", or "invalid response”. | C3 | Response with the weather details of mentioned city name as string, or error message consider to the situation. |

* **We put which methods relevant for each characteristic :**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **ID** | **Character -istic** | **getCurrentWeather**  **(CityName)** | **getCurrentWeather**  **(CityId)** | **getCurrentWeather**  **(Coordinates)** | **Partitions** |
| C1 | Response with the weather details of mentioned city name as string, or error message consider to the situation. | x |  |  | {true,false} |
| C2 | Response with the weather details of mentioned city name as string, or error message consider to the situation. |  | x |  | {true,false} |
| C3 | Response with the weather details of mentioned city name as string, or error message consider to the situation. |  |  | x | {true,false} |

**Class#2**

**(GsonService)**

* **Method1(fromJson(String jsonString, Class<T> classOfT))):**
* **Partitions**:

**\*For string jsonString we have 1 partitions:**

* A1:take the response from getCurrentWeather(Correct response)
* A2:take the response from getCurrentWeather(Wrong response)

**\*For Class<T> classOfT we have 1 partitions:**

* B1:class that wants its partitions

\***For output :**

* **C1**: return response as partitions
* **The chosen coverage criteria with complete explanation:**
* we used ACOC because we had used all the combination of partitions from all characteristics, that results 2 test Cases :
* t1: **jsonString=Correct response, classOfT** =class name(covers A1,B1,C1)
* t2: **jsonString=Wrong response, classOfT** =class name(covers A2,B1,C1)
* **Method2(toJson(Object object)):**
* **Partitions**:

**\*For Object object we have 4 partitions:**

* B1:object response with each partitions as a whole(Correct response)
* B2: object response with each partitions as a whole(Wrong response)

\***For output :**

* **C1**: response as string
* **The chosen coverage criteria with complete explanation:**
* we used ACOC because we had used all the combination of partitions from all characteristics, that results 2 test Cases :
* t1: **object** =Correct response (covers B1,C1)
* t2: **object** =Wrong response (covers B2,C1)
* **Report for all testcases**:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Method** | **Params** | **Returns** | **Values** | **Ch ID** | **Character -istic** |
| **fromJson** | String jsonString, Class<T> classOfT | <T> T | return response as partitions. | C1 | return response as partitions with there details. |
| **toJson** | Object object | string | response as string. | C2 | response as string. |

* **We put which methods relevant for each characteristic :**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **ID** | **Character -istic** | **fromJson** | **toJson** | **Partitions** |
| C1 | return response as partitions with there details. | x |  | {true,false} |
| C2 | response as string |  | x | {true,false} |

**Class#3**

**(WeatherService)**

* **Method1(getCurrentWeather(String cityName)):**
* **Partitions**:

**\*For string cityName we have 4 partitions:**

* B1:empty string
* B2:null string
* B3:wrong city name
* B4:non empty string

\***For output :**

* **C1**: invalid response
* **C2:** Actual response
* **Boundaries**:

\***For string cityName we have**

* There are no upper bounds for s and txt (i.e., no upper limit on their length).(valid) and The lower bound on their length is 0, or null .(valid)
* **The chosen coverage criteria with complete explanation:**
* we used ACOC because we had used all the combination of partitions from all characteristics, that results 4 test Cases :
* t1: **cityName** =” “ (covers B1,C1)
* t2: **cityName =**null (covers B2,C1)
* t3: **cityName =** wrong city name”lodon” (covers B3,C1)
* t4: **cityName** = non empty string “London”(covers B4,C2)
* **Method2(getCurrentWeather(int cityId)):**
* **Partitions**:

**\*For int cityId we have 4 partitions:**

* B1:negative(<0)
* B2:equals 0
* B3:positive(>0)(id with no city)
* B4:positive(>0)(id with city)

\***For output :**

* **C1**: invalid response
* **C2:** Actual response
* **Boundaries**:

\***For integer CityId**

* valid for n>0(Correct response), n<=0(Wrong response)
* **The chosen coverage criteria with complete explanation:**
* we used ACOC because we had used all the combination of partitions from all characteristics, that results 4 test Cases :
* t1: **cityId** =negative(-2) (covers B1,C1)
* t2: **cityId =**0(covers B2,C1)
* t3: **cityId =**800(covers B3,C1)
* t4: **cityId =** positive(2643743) (covers B4,C2)
* **Method3(getCurrentWeather(double latitude, double longitude)):**
* **Partitions**:

**\*For double latitude we have 3 partitions:**

* A1:negative(<0)
* A2:equals 0
* A3:positive(>0)

**\*For double longitude we have 3 partitions:**

* B1:negative(<0)
* B2:equals 0
* B3:positive(>0)

\***For output :**

* **C1**: invalid response
* **C2:** Actual response
* **Boundaries**:

\***For double** latitude/ longitude

* valid for latitude >/</= 0 ,longitude >/</=0
* **The chosen coverage criteria with complete explanation:**
* We used BCC because we choose to use the base choice for each characteristic, that results 5 testCases:
* t1: latitude =negative(-0.1257), longitude =positive (51.5085)(base case/happy path) (covers A1 ,B3,C2)
* t2: latitude =negative(-1), longitude =negative (-1) (covers A1 ,B1,C1)
* t3: latitude =negative(-0.1257), longitude =zero (0) (covers A1 ,B2,C2)
* t4: latitude =zero(0), longitude =zero (0) (covers A1 ,B2,C2)
* t5: latitude =positive(0.1257), longitude =zero (0) (covers A3 ,B2,C2)
* **Report for all testcases**:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Method** | **Params** | **Returns** | **Values** | **Ch ID** | **Character -istic** |
| **getCurrentWeather** | String  CityName | string | Response with the weather details of mentioned city name as string, or invalid response. | C1 | Response with the weather details of mentioned city name as string, or error message consider to the situation. |
| **getCurrentWeather** | String  CityId | string | Response with the weather details of mentioned city name as string, or invalid response. | C2 | Response with the weather details of mentioned city name as string, or error message consider to the situation. |
| **getCurrentWeather** | double latitude, double longitude | string | Response with the weather details of mentioned city name as string, or invalid response. | C3 | Response with the weather details of mentioned city name as string, or error message consider to the situation. |

* **We put which methods relevant for each characteristic :**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **ID** | **Character -istic** | **getCurrentWeather**  **(CityName)** | **getCurrentWeather**  **(CityId)** | **getCurrentWeather**  **(Coordinates)** | **Partitions** |
| C1 | Response with the weather details of mentioned city name as string, or error message consider to the situation. | x |  |  | {true,false} |
| C2 | Response with the weather details of mentioned city name as string, or error message consider to the situation. |  | x |  | {true,false} |
| C3 | Response with the weather details of mentioned city name as string, or error message consider to the situation. |  |  | x | {true,false} |