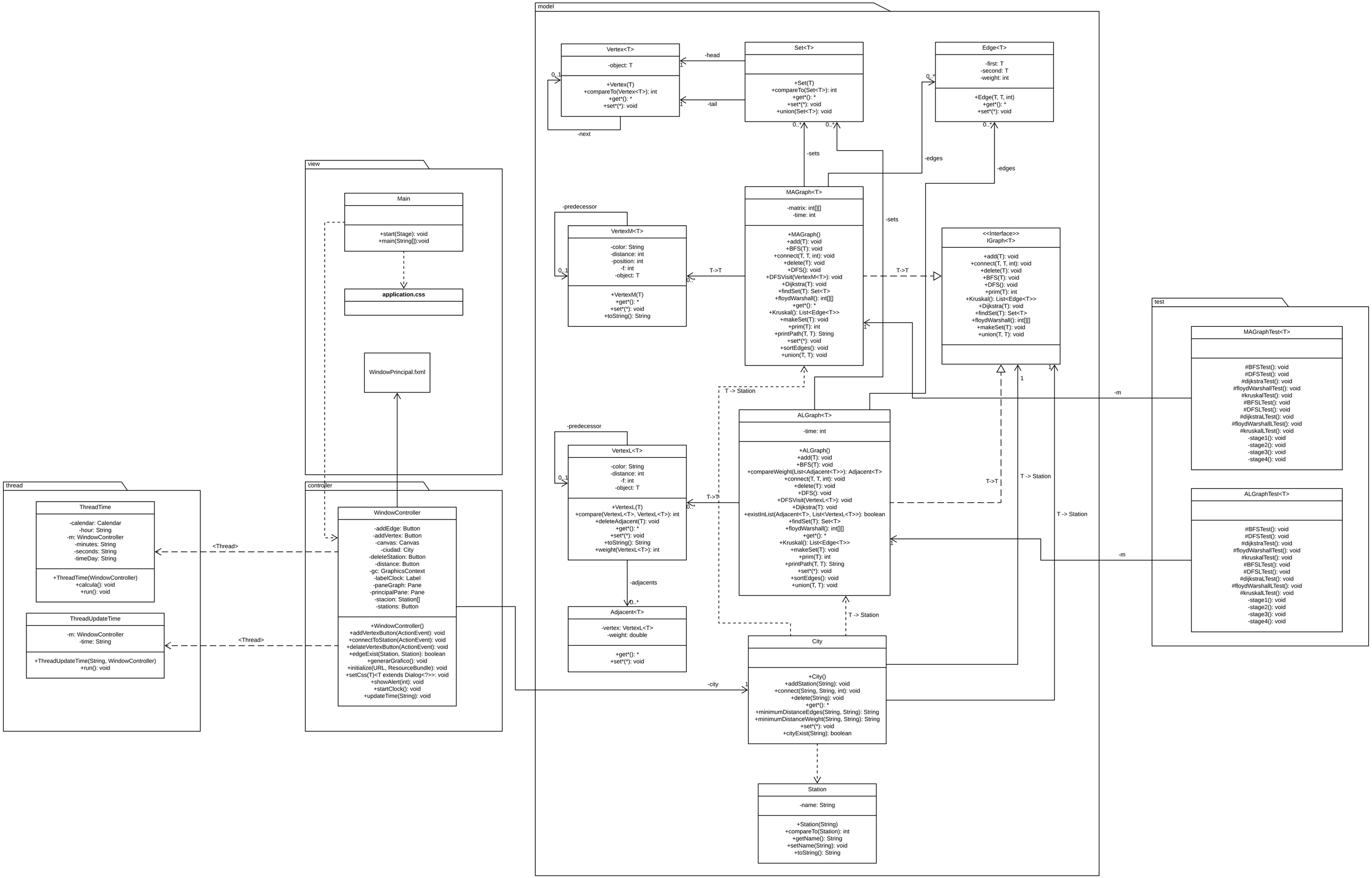
model
Vertex<T>
-object: T
+Vertex(T)
+compareTo(Vertex<T>): int
+get*(): *
+set*(*): void
-next
0..1
-head
-tail
1
Set<T>
+Set(T)
+compareTo(Set<T>): int
+get*(): *
+set*(*): void
+union(Set<T>): void
Edge<T>
-first: T
-second: T
-weight: int
+Edge(T, T, int)
+get*(): *
+set*(*): void
0..*
-sets
-edges
MAGraph<T>
-matrix: int[][]
-time: int
+MAGraph()
+add(T): void
+BFS(T): void
+connect(T, T, int): void
+delete(T): void
+DFS(): void
+DFSVisit(VertexM<T>): void
+Dijkstra(T): void
+findSet(T): Set<T>
+floydWarshall(): int[][]
+get*(): *
+Kruskal(): List<Edge<T>>
+makeSet(T): void
+prim(T): int
+printPath(T, T): String
+set*(*): void
+sortEdges(): void
+union(T, T): void
-predecessor
VertexM<T>
-color: String
-distance: int
-position: int
-f: int
-object: T
+VertexM(T)
+get*(): *
+set*(*): void
+toString(): String
T->T
<<Interface>>
IGraph<T>
+add(T): void
+connect(T, T, int): void
+delete(T): void
+BFS(T): void
+DFS(): void
+prim(T): int
+Kruskal(): List<Edge<T>>
+Dijkstra(T): void
+findSet(T): Set<T>
+floydWarshall(): int[][]
+makeSet(T): void
+union(T, T): void
1
T -> Station
ALGraph<T>
-time: int
+ALGraph()
+add(T): void
+BFS(T): void
+compareWeight(List<Adjacent<T>>): Adjacent<T>
+connect(T, T, int): void
+delete(T): void
+DFS(): void
+DFSVisit(VertexL<T>): void
+Dijkstra(T): void
+existInList(Adjacent<T>, List<VertexL<T>>): boolean
+findSet(T): Set<T>
+floydWarshall(): int[][]
+get*(): *
+Kruskal(): List<Edge<T>>
+makeSet(T): void
+prim(T): int
+printPath(T, T): String
+set*(*): void
+sortEdges(): void
+union(T, T): void
-predecessor
VertexL<T>
-color: String
-distance: int
-f: int
-object: T
+VertexL(T)
+compare(VertexL<T>, VertexL<T>): int
+deleteAdjacent(T): void
+get*(): *
+set*(*): void
+toString(): String
+weight(VertexL<T>): int
-adjacents
Adjacent<T>
-vertex: VertexL<T>
-weight: double
+get*(): *
+set*(*): void
City
+City()
+addStation(String): void
+connect(String, String, int): void
+delete(String): void
+get*(): *
+minimumDistanceEdges(String, String): String
+minimumDistanceWeight(String, String): String
+set*(*): void
+cityExist(String): boolean
-city
Station
-name: String
+Station(String)
+compareTo(Station): int
+getName(): String
+setName(String): void
+toString(): String
-m
test
MAGraphTest<T>
#BFSTest(): void
#DFSTest(): void
#dijkstraTest(): void
#floydWarshallTest(): void
#kruskalTest(): void
#BFSLTest(): void
#DFSLTest(): void
#dijkstraLTest(): void
#floydWarshallLTest(): void
#kruskalLTest(): void
-stage1(): void
-stage2(): void
-stage3(): void
-stage4(): void
ALGraphTest<T>
#BFSTest(): void
#DFSTest(): void
#dijkstraTest(): void
#floydWarshallTest(): void
#kruskalTest(): void
#BFSLTest(): void
#DFSLTest(): void
#dijkstraLTest(): void
#floydWarshallLTest(): void
#kruskalLTest(): void
-stage1(): void
-stage2(): void
-stage3(): void
-stage4(): void
view
Main
+start(Stage): void
+main(String[]):void
application.css
WindowPrincipal.fxml
thread
ThreadTime
-calendar: Calendar
-hour: String
-m: WindowController
-minutes: String
-seconds: String
-timeDay: String
+ThreadTime(WindowController)
+calcula(): void
+run(): void
ThreadUpdateTime
-m: WindowController
-time: String
+ThreadUpdateTime(String, WindowController)
+run(): void
<Thread>
controler
WindowController
-addEdge: Button
-addVertex: Button
-canvas: Canvas
-ciudad: City
-deleteStation: Button
-distance: Button
-gc: GraphicsContext
-labelClock: Label
-paneGraph: Pane
-principalPane: Pane
-stacion: Station[]
-stations: Button
+WindowController()
+addVertexButton(ActionEvent): void
+connectToStation(ActionEvent): void
+delateVertexButton(ActionEvent): void
+edgeExist(Station, Station): boolean
+generarGrafico(): void
+initialize(URL, ResourceBundle): void
+setCss(T)<T extends Dialog<?>>: void
+showAlert(int): void
+startClock(): void
+updateTime(String): void