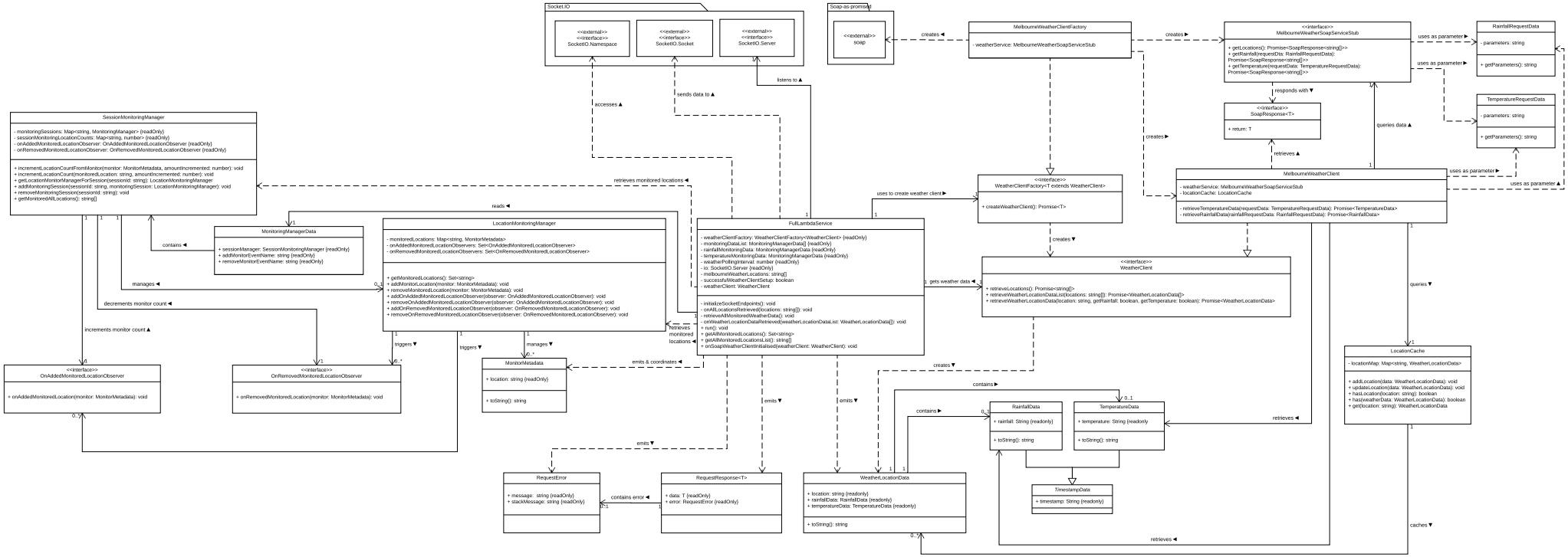
Socket.IO
<<external>>
<<interface>>
SocketIO.Namespace
<<external>>
<<interface>>
SocketIO.Socket
<<external>>
<<interface>>
SocketIO.Server
Soap-as-promised
<<external>>
soap
creates ◀
MelbourneWeatherClientFactory
- weatherService: MelbourneWeatherSoapServiceStub
creates ▶
<<interface>>
MelbourneWeatherSoapServiceStub
+ getLocations(): Promise<SoapResponse<string[]>>
+ getRainfall(requestDta: RainfallRequestData): Promise<SoapResponse<string[]>>
+ getTemperature(requestData: TemperatureRequestData): Promise<SoapResponse<string[]>>
uses as parameter ▶
RainfallRequestData
- parameters: string
+ getParameters(): string
uses as parameter ▶
responds with ▼
<<interface>>
SoapResponse<T>
+ return: T
TemperatureRequestData
- parameters: string
+ getParameters(): string
listens to ▲
accesses ▲
sends data to ▲
SessionMonitoringManager
- monitoringSessions: Map<string, MonitoringManager> {readOnly}
- sessionMonitoringLocationCounts: Map<string, number> {readOnly}
- onAddedMonitoredLocationObserver: OnAddedMonitoredLocationObserver {readOnly}
- onRemovedMonitoredLocationObserver: OnRemovedMonitoredLocationObserver {readOnly}
+ incrementLocationCountFromMonitor(monitor: MonitorMetadata, amountIncremented: number): void
+ incrementLocationCount(monitoredLocation: string, amountIncremented: number): void
+ getLocationMonitorManagerForSession(sessionId: string): LocationMonitoringManager
+ addMonitoringSession(sessionId: string, monitoringSession: LocationMonitoringManager): void
+ removeMonitoringSession(sessionId: string): void
+ getMonitoredAllLocations(): string[]
retrieves monitored locations ◀
creates ▶
queries data ▲
retrieves ▲
<<interface>>
WeatherClientFactory<T extends WeatherClient>
+ createWeatherClient(): Promise<T>
uses to create weather client ▶
MelbourneWeatherClient
- weatherService: MelbourneWeatherSoapServiceStub
- locationCache: LocationCache
- retrieveTemperatureData(requestData: TemperatureRequestData): Promise<TemperatureData>
- retrieveRainfallData(rainfallRequestData: RainfallRequestData): Promise<RainfallData>
uses as parameter ▶
uses as parameter ▲
reads ◀
MonitoringManagerData
+ sessionManager: SessionMonitoringManager {readOnly}
+ addMonitorEventName: string {readOnly}
+ removeMonitorEventName: string {readOnly}
contains ◀
LocationMonitoringManager
- monitoredLocations: Map<string, MonitorMetadata>
- onAddedMonitoredLocationObservers: Set<OnAddedMonitoredLocationObserver>
- onRemovedMonitoredLocationObservers: Set<OnRemovedMonitoredLocationObserver>
+ getMonitoredLocations(): Set<string>
+ addMonitorLocation(monitor: MonitorMetadata): void
+ removeMonitoredLocation(monitor: MonitorMetadata): void
+ addOnAddedMonitoredLocationObserver(observer: OnAddedMonitoredLocationObserver): void
+ removeOnAddedMonitoredLocationObserver(observer: OnAddedMonitoredLocationObserver): void
+ addOnRemovedMonitoredLocationObserver(observer: OnRemovedMonitoredLocationObserver): void
+ removeOnRemovedMonitoredLocationObserver(observer: OnRemovedMonitoredLocationObserver): void
manages ◀
decrements monitor count ◀
increments monitor count ▲
FullLambdaService
- weatherClientFactory: WeatherClientFactory<WeatherClient> {readOnly}
- monitoringDataList: MonitoringManagerData[] {readOnly}
- rainfallMonitoringData: MonitoringManagerData {readOnly}
- temperatureMonitoringData: MonitoringManagerData {readOnly}
- weatherPollingInterval: number {readOnly}
- io: SocketIO.Server {readOnly}
- melbourneWeatherLocations: string[]
- successfulWeatherClientSetup: boolean
- weatherClient: WeatherClient
- initializeSocketEndpoints(): void
- onAllLocationsRetrieved(locations: string[]): void
- retrieveAllMonitoredWeatherData(): void
- onWeatherLocationDataRetrieved(weatherLocationDataList: WeatherLocationData[]): void
+ run(): void
+ getAllMonitoredLocations(): Set<string>
+ getAllMonitoredLocationsList(): string[]
+ onSoapWeatherClientInitialised(weatherClient: WeatherClient): void
gets weather data ◀
creates ▼
<<interface>>
WeatherClient
+ retrieveLocations(): Promise<string[]>
+ retrieveWeatherLocationDataList(locations: string[]): Promise<WeatherLocationData[]>
+ retrieveWeatherLocationData(location: string, getRainfall: boolean, getTemperature: boolean): Promise<WeatherLocationData>
queries ▼
LocationCache
- locationMap: Map<string, WeatherLocationData>
+ addLocation(data: WeatherLocationData): void
+ updateLocation(data: WeatherLocationData): void
+ hasLocation(location: string): boolean
+ has(weatherData: WeatherLocationData): boolean
+ get(location: string): WeatherLocationData
<<interface>>
OnAddedMonitoredLocationObserver
+ onAddedMonitoredLocation(monitor: MonitorMetadata): void
<<interface>>
OnRemovedMonitoredLocationObserver
+ onRemovedMonitoredLocation(monitor: MonitorMetadata): void
triggers ▼
triggers ▼
manages ▼
MonitorMetadata
+ location: string {readOnly}
+ toString(): string
retrieves monitored locations ◀
emits & coordinates ◀
creates ▼
contains ▶
contains ▶
RainfallData
+ rainfall: String {readonly}
+ toString(): string
TemperatureData
+ temperature: String {readonly
+ toString(): string
retrieves ◀
TimestampData
+ timestamp: String {readonly}
emits ▼
emits ▼
emits ▼
RequestError
+ message: string {readOnly}
+ stackMessage: string {readOnly}
contains error ◀
RequestResponse<T>
+ data: T {readOnly}
+ error: RequestError {readOnly}
WeatherLocationData
+ location: string {readonly}
+ rainfallData: RainfallData {readonly}
+ temperatureData: TemperatureData {readonly}
+ toString(): string
retrieves ◀
caches ▼