## What is JSON Graph?

JSON Graph is a convention for modeling graph information as a JSON object.

Applications that use Falcor represent all their domain data as a single JSON Graph object.

JSON Graph is valid JSON and can be parsed by any JSON parser.

However JSON Graph introduces new primitive types to JSON to allow JSON to be used to represent graph information in a simple and consistent way. While it is possible to work with JSON Graph data directly, it is much more convenient to work with it indirectly using tools that understand JSON Graph types. One way to think about Falcor is as a set of protocols and tools for transferring, storing, and retrieving data from JSON Graph objects.

Here’s a simple example of a JSON Graph Object that represents a TODO list where each task can be linked to one or more prerequisites.

{

todosById: {

"44": {

name: "get milk from corner store",

done: false,

prerequisites: [{ $type: "ref", value: ["todosById", 54] }]

},

"54": {

name: "withdraw money from ATM",

done: false,

prerequisites: []

}

},

todos: [

{ $type: "ref", value: ["todosById", 44] },

{ $type: "ref", value: ["todosById", 54] }

]

};

Note that in the example above the JSON Graph contains references to other locations in the same object. It is this concept of a Reference that allows graphs to be represented in JSON.

## Why JSON Graph?

JSON is a ubiquitous data interchange format. Web applications often exchange data in JSON format because manipulating JSON Data in JavaScript is so easy. JSON is also map–based, which makes it easy to divide a large data set into smaller subsets and send them across the wire on demand.

Unfortunately there is a downside to using JSON to send and store your Web application’s data: JSON models trees, and most application domains are graphs. As a result, serializing a graph as JSON can introduce duplicate copies of the same entity.

## How Does JSON Graph Work?

## How Does JSON Graph Work?

**JSON Graph allows a graph to be modeled as JSON without introducing duplicates.** Instead of inserting an entity into the same message multiple times, each entity with a unique identifier is inserted into a single, globally unique location in the JSON Graph object. The [Path](http://netflix.github.io/falcor/documentation/paths.html) to the only location within the JSON Graph object where an entity is stored is referred to as the entity’s **Identity Path.** No two entities in an application’s domain model should have the same **Identity Path**. If an entity’s unique identifier (often assigned by a data store) is not globally unique, but rather only unique within the set of its like types, the entity’s **Identity Path** can be changed to include both the its type and ID. This combination is usually enough to ensure that an entity’s **Identity Path** is globally unique.

Whenever an entity needs to be referenced by another entity in the same JSON graph object, a**Reference** with the entity’s **Identity Path** is included instead. A Reference is a new value type that JSON Graph introduces to JSON to allow graph relationships to be modeled within a JSON object

https://netflix.github.io/falcor/