## 2. What is a Resource?

The key **abstraction of information** in REST is a [resource](https://restfulapi.net/resource-naming/). Any information that we can name can be a resource. For example, a REST resource can be a document or image, a temporal service, a collection of other resources, or a non-virtual object (e.g., a person).

The state of the resource, at any particular time, is known as the **resource representation**.

The resource representations are consist of:

* the **data**
* the **metadata** describing the data
* and the **hypermedia links** that can help the clients in transition to the next desired state.

A REST API consists of an assembly of interlinked resources. This set of resources is known as the REST API’s ***resource model***.

### 2.1. Resource Identifiers

REST uses resource identifiers to identify each resource involved in the interactions between the client and the server components.

### 2.2. Hypermedia

The data format of a representation is known as a [media type](https://www.iana.org/assignments/media-types/media-types.xhtml). The media type identifies a specification that defines how a representation is to be processed.

**A RESTful API looks like**[***hypertext***](https://restfulapi.net/hateoas/)**.** Every addressable unit of information carries an address, either explicitly (e.g., link and id attributes) or implicitly (e.g., derived from the media type definition and representation structure).

*Hypertext* (or hypermedia) means the **simultaneous presentation of information and controls** such that the information becomes the affordance through which the user (or automaton) obtains choices and selects actions.

Remember that hypertext does not need to be HTML (or XML or JSON) on a browser. Machines can follow links when they understand the data format and relationship types.

— Roy Fielding

### 2.3. Self-Descriptive

Further, **resource representations shall be self-descriptive**: the client does not need to know if a resource is an employee or a device. It should act based on the media type associated with the resource.

So in practice, we will create lots of **custom media types** – usually one media type associated with one resource.

Every media type defines a default processing model. For example, HTML defines a rendering process for hypertext and the browser behavior around each element.

Media Types have no relation to the resource methods GET/PUT/POST/DELETE/… other than the fact that some media type elements will define a process model that goes like “anchor elements with an href attribute create a hypertext link that, when selected, invokes a retrieval request (GET) on the URI corresponding to the CDATA-encoded href attribute.”

HELLO ZIBA