What is the Document Object Model?

Introduction

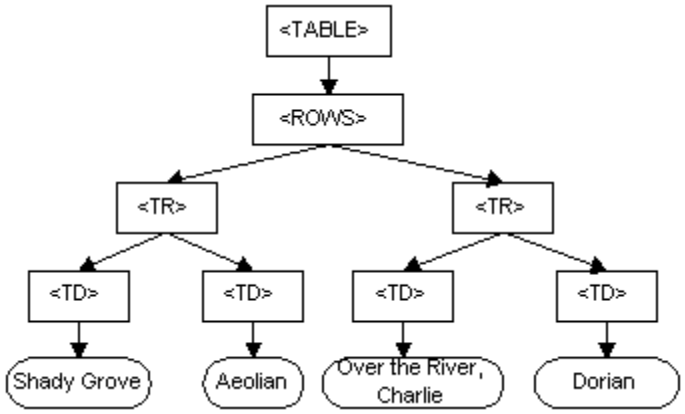
- The Document Object Model (DOM) is a programming API for HTML and XML documents. It defines the logical structure of documents and the way a document is accessed and manipulated.
- In the DOM specification, the term "document" is used in the broad sense - increasingly, XML is being used as a way of representing many different kinds of information that may be stored in diverse systems, and much of this would traditionally be seen as data rather than as documents.
- Nevertheless, XML presents this data as documents, and the DOM may be used to manage this data.
- With the Document Object Model, programmers can create and build documents, navigate their structure, and add, modify, or delete elements and content.
- Anything found in an HTML or XML document can be accessed, changed, deleted, or added using the Document Object Model
- As a W3C specification, one important objective for the Document Object Model is to provide a standard programming interface that can be used in a wide variety of environments and applications.
- The Document Object Model can be used with any programming language.

What the Document Object Model is

The Document Object Model is a programming API for documents. The object model itself closely resembles the structure of the documents it models. For instance, consider this table, taken from an HTML document:

```
<TABLE>
<ROWS>
<TR>
<TD>Shady Grove</TD>
<TD>Aeolian</TD>
</TR>
<TR>
<TD>Over the River, Charlie</TD>
<TD>Dorian</TD>
</TR>
</ROWS>
</TABLE>
```

The Document Object Model represents this table like this:



DOM representation of the example table

- In the Document Object Model, documents have a logical structure which is very much like a tree; to be more precise, it is like a "forest" or "grove" which can contain more than one tree. However, the Document Object Model does not specify that documents be *implemented* as a tree or a grove , nor does it specify how the relationships among objects be implemented in any way.
- In other words, the object model specifies the logical model for the programming interface, and this logical model may be implemented in any way that a particular implementation finds convenient.
- In this specification, we use the term *structure model* to describe the tree-like representation of a document; we specifically avoid terms like "tree" or "grove" in order to avoid implying a particular implementation.
- One important property of DOM structure models is *structural isomorphism*: if any two Document Object Model implementations are used to create a representation of the same document, they will create the same structure model, with precisely the same objects and relationships.
- As an object model, the Document Object Model identifies:

    - the interfaces and objects used to represent and manipulate a document
    - the semantics of these interfaces and objects - including both behavior and attributes
    - the relationships and collaborations among these interfaces and objects

- All DOM implementations must support the interfaces listed as "fundamental" in the Core specification; in addition, XML implementations must support the interfaces listed as "extended" in the Core specification.
- The Level 1 DOM HTML specification defines additional functionality needed for HTML documents.

What the Document Object Model is not

This section is designed to give a more precise understanding of the Document Object Model by distinguishing it from other systems that may seem to be like it.

- The Document Object Model is not a binary specification. Document Object Model programs written in the same language will be source code compatible across platforms, but the Document Object Model does not define any form of binary interoperability.
- The Document Object Model is not a way of persisting objects to XML or HTML. Instead of specifying how objects may be represented in XML, the Document Object Model specifies how XML and HTML documents are represented as objects, so that they may be used in object oriented programs.
- The Document Object Model is not a set of data structures, it is an object model that specifies interfaces.
- The Document Object Model does not define "the true inner semantics" of XML or HTML. The semantics of those languages are defined by the languages themselves. The Document Object Model is a programming model designed to respect these semantics.