

Service Data Objects for PHP

Service Data Objects (SDOs) enable PHP applications to work with data from different sources, such as databases and XML files, using a single interface.

The PHP implementation of Service Data Objects is the first to map SDO to a weakly and dynamically typed scripting environment. The result is a technology which, at its heart provides the same functionality and compatibility as other implementations, but externally present an API which is tailored to the target language.

For example, SDO in the strongly and statically typed languages of Java and C++ has many get/set methods for accessing data object properties of various types, such as float, string, int, etc. In PHP, no such distinction is required because, where necessary, the runtime juggles source types to match their target ([see \[4\]](#)).

SDO also exploits PHP's support for dynamic interfaces ([see \[5\]](#)), which PHP calls "method overloading". This allows a class to support methods for which it has no explicit definition. SDO uses this capability to present an interface to the developer which closely reassembles the data structure being manipulated. In statically typed languages such as Java and C++, the developer is required to generate code for an interface to achieve the same effect. Let's consider the example of the AccountSummary from the Service Component Architecture BigBank scenario ([see \[6\]](#)). This has a schema definition of the following:

```
<xsd:complexType name="AccountSummary">
  <xsd:sequence>
    <xsd:element name="accountNumber" type="xsd:string"/>
    <xsd:element name="accountType" type="xsd:string"/>
    <xsd:element name="balance" type="xsd:float"/>
  </xsd:sequence>
</xsd:complexType>
```

A PHP SDO initialized with this schema definition using the XML Data Access Service (see <http://www.php.net/sdo-das-xml>) would automatically allow us to get and set the data with the following calls.

```
$accountSummary->accountNumber = "1234 1234 1234 1234";
$accountSummary->accountType = "Checking";
$accountSummary->balance=100.10;
echo $accountSummary->accountNumber;
echo $accountSummary->accountType;
echo $accountSummary->balance;
```

Data Access Services

SDO for PHP provides two Data Access Service (DAS) implementations; the XML Data Access Service (XML DAS) for working with XML data, and the Relational Data Access Service (RDAS) for working with databases.

The XML DAS creates SDOs based off an XML Schema definition for the data structure. It can be used to load and save SDOs to and from XML instance documents or a URL. The combination of SDO and the XML DAS provides a very simple way of navigating and reading XML documents, but its great strength is in its ability to also create XML documents through the same simple API (see <http://www.ibm.com/developerworks/opensource/library/os-php-sdo/> for an article on building a simple RSS feed).

The RDAS creates SDOs based off a definition of the database schema. In addition to creating new SDOs, the RDAS supports retrieval of existing data from a relational database into a graph of SDOs. This graph can then be modified; values changed, data objects created or deleted, and then these changes can be applied back to the database through the RDAS.

The RDAS implements the Optimistic Offline-Lock pattern (see <http://www.martinfowler.com/eaaCatalog/optimisticOfflineLock.html>), which makes it ideal for many Web scenarios where data is 'checked out' for editing, but not locked and then updates are applied some time later. The RDAS ensures that any database concurrency conflicts are detected and reported.

Next steps

One of the key strengths of SOA is interoperability. Scenarios will demonstrate interworking between SDO for PHP and other implementations, exchanging not only the SDO data but also its change history, enabling cross-platform management and update of the data.

PHP is used in a wide range of environments where services may provide data conforming to a wide variety of formats, perhaps including RSS, ATOM, XML databases, RelaxNG and many more. The SDO for PHP interfaces allow new Data Access Services to be developed which can simplify the exchange of data with these services and unify the programming interfaces to the data.

Resources

[1] The PHP implementation of the Service Data Objects technology was first made available in July 2005. It ships as a native extension to the PHP runtime and can be downloaded from the PECL repository

<http://pecl.php.net/sdo>

[2] SDO for PHP and the Data Access Services are documented as part of the PHP manual

<http://www.php.net/docs>

<http://www.php.net/sdo>

[3] There are a number of articles on SDO for PHP available

<http://www.ibm.com/developerworks/opensource/library/os-php-sdo/>
<http://www.ibm.com/developerworks/library/os-sdphp/>

[4] Information on PHP type juggling

<http://www.php.net/language.types.type-juggling>

[5] Information on dynamic interfaces in PHP

<http://www.php.net/language.oop5.overloading>

[6] SCA BigBank Example

http://download.boulder.ibm.com/ibmdl/pub/software/dw/specs/ws-sca/SCA_BuildingYourFirstApplication_V09.pdf