

HP OpenView Service Desk 4.5

Web API Programmer's Guide

First Edition



i n v e n t

Manufacturing Part Number: N/A

July 2002

Legal Notices

Hewlett-Packard makes no warranty of any kind with regard to this manual, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. Hewlett-Packard shall not be held liable for errors contained herein or direct, indirect, special, incidental or consequential damages in connection with the furnishing, performance, or use of this material.

Restricted Rights Legend. Use, duplication or disclosure by the U.S. Government is subject to restrictions as set forth in subparagraph (c)(1)(ii) of the Rights in Technical Data and Computer Software clause in DFARS 252.227-7013.

Hewlett-Packard Company
3000 Hanover Street
Palo Alto, CA 94304 U.S.A.

Rights for non-DOD U.S. Government Departments and Agencies are as set forth in FAR 52.227-19 (c)(1,2).

Copyright Notice. © Copyright 2002 Hewlett-Packard Company

The nomenclature of each version of this software (and manuals therefore) has been devised for commercially convenient reasons, and is not intended to denote the degree of originality of any version of the software with respect to any other version. The extent of protection afforded by, and duration of copyright is to be determined entirely independently of this nomenclature.

Trademark Notices

Java™ is a U.S. trademark of Sun Microsystems, Inc.

Microsoft® is a U.S. registered trademark of Microsoft Corporation.

Oracle® is a registered U.S. trademark of Oracle Corporation, Redwood City, California.

UNIX® is a registered trademark of the Open Group.

Windows NT® is a U.S. registered trademark of Microsoft Corporation.

Windows® and MS Windows® are U.S. registered trademarks of Microsoft Corporation.

1. Installation

Class files	22
Overview	22
Usage	22

2. API Structure

Structure Outline	24
Server- and Client-Side Components	24
Client-Side Structure	24

3. Overview of Available Methods

Methods in Home Classes	30
Open method	30
Open new method	30
Find methods	30
Search methods	31
Methods in Entity Classes	32
Non-aggregated entities:	32
Aggregated entities:	32
Utility Classes	33
Entitlement Classes	33
Where Building Classes	33
addCriteriumOn<attribute name>(value),	33
addCriteriumOn<attribute name>Range(arg[])	33
addContainCriteriumOn<attribute name>(String value)	34
Getting and Setting Attribute Values	35
Getting and Setting Attributes of Basic Types	35
Getting and Setting Entity Reference Attributes	36
Handling Entity Set Reference Attributes	36
Handling n-m Relation Attributes	37
Java Comment for Attribute Methods	37
Reverse Get Methods	38
Web API Application	39
The Web API Javadoc	40
Examples	41
Example1.java	41
Example2.java	41
Example3.java	41

Contents

Example4.java.	41
Example5.java.	41
Example6.java.	41
Example7.java.	42

4. Appendix

Changes from Service Desk 4.0 to 4.5.	44
--	----

Figure 2-1. Entity structure	26
Figure 2-2. Home Structure.....	27

Preface

This document describes procedures for using the Web API for Service Desk. The Web API can be used to create customized integrations with Service Desk.

This guide is intended for use by any application administrator.

This guide is organized as follows:

- Chapter 1, “Installation,” on page 21 explains how to install the Web API.
- Chapter 2, “API Structure,” on page 23 describes the structure of the API.
- Chapter 3, “Overview of Available Methods,” on page 29 provides an overview of the methods available in the API.
- Chapter 4, “Appendix,” on page 43 explains what has changed since the Service Desk 4.0 version.

Revision History

When an edition of a manual is issued with a software release, it has been reviewed and tested and is therefore considered correct at the date of publication. However, errors in the software or documentation that were unknown at the time of release, or important new developments, may necessitate the release of a service pack that includes revised documentation. Revised documentation may also be published on the Internet, see “We Welcome Your Comments!” in this preface for the URL.

A revised edition will display change bars in the left-hand margin to indicate revised text. These change bars will only mark the text that has been edited or inserted since the previous edition or revised edition.

When a revised edition of this document is published, the latest revised edition nullifies all previous editions.

Table 1

Revision History

Edition and Revision Number	Issue Date	Product Release
First Edition	July 2002	Service Desk 4.5

Related Publications

This section helps you find information that is related to the information in this guide. It gives an overview of the Service Desk documentation and lists other publications you may need to refer to when using this guide.

The Service Desk Documentation

Service Desk provides a selection of books and online help to assist you in using Service Desk and improve your understanding of the underlying concepts. This section illustrates what information is available and where you can find it.

NOTE

This section lists the publications provided with Service Desk 4.5. Updates of publications and additional publications may be provided in later service packs. For an overview of the documentation provided in service packs, please refer to the readme file of the latest service pack. The service packs and the latest versions of publications are available on the Internet, at <http://support.openview.hp.com/cpe/patches> and http://ovweb.external.hp.com/lpe/doc_serv respectively. See the section “We Welcome Your Comments!” in this preface for the URLs.

-
- The `Readme.htm` files on the Service Desk CD-ROMs contain information that will help you get started with Service Desk. The Readme files also contain any last-minute information that became available after the other documentation went to manufacturing.

The Service Desk 4.5 server is coded in Pure Java and is platform independent. The installation software for each platform varies. Service Desk is therefore distributed on three CD-ROMs, one each for Microsoft Windows (2000 and NT4), HP-UX, and Sun Solaris. A different readme file is available on each CD-ROM.

- The *HP OpenView Service Desk: Release Notes* give a description of the features that Service Desk provides. In addition, they give information that helps you:
 - compare the current software’s features with those available in previous versions of the software;
 - solve known problems.

The Release Notes are available as a PDF file on the HP OpenView Service Desk 4.5 for Windows CD-ROM . The file name is `Release_Notes.pdf`.

- The *HP OpenView Service Desk: User's Guide* introduces you to the key concepts behind Service Desk. It gives an overview of what you can do with Service Desk and explains typical tasks of different types of Service Desk users. Scenario descriptions are provided as examples of how the described features could be implemented.

The User's Guide is available as a PDF file on the HP OpenView Service Desk 4.5 for Windows CD-ROM. The file name is `User's_Guide.pdf`.

- The *HP OpenView Service Desk: Supported Platforms List* contains information that helps you determine software requirements. It lists the software versions supported by Hewlett-Packard for Service Desk 4.5.

The Supported Platforms List is available as a PDF file on the HP OpenView Service Desk 4.5 for Windows CD-ROM. The file name is `Supported_Platforms_List.pdf`.

- The *HP OpenView Service Desk: Installation Guide* covers all aspects of installing Service Desk.

The Installation Guide is available as a PDF file on the HP OpenView Service Desk 4.5 for Windows CD-ROM. The file name is `Installation_Guide.pdf`.

- The *HP OpenView Service Desk: Administrator's Guide* provides information that helps application administrators to set up and maintain the Service Desk application server for client usability.

The Administrator's Guide is available as a PDF file on the HP OpenView Service Desk 4.5 for Windows CD-ROM. The file name is `Administrator's_Guide.pdf`.

- The *HP OpenView Service Desk: Data Exchange Administrator's Guide* explains the underlying concepts of the data exchange process and gives instructions on exporting data from external applications and importing it into Service Desk. The data exchange process includes importing single service events and batches of data.

The Data Exchange Administrator's Guide is available as a PDF file on the HP OpenView Service Desk 4.5 for Windows CD-ROM. The file name is `Data_Exchange.pdf`.

- The *HP OpenView Operations Integration Administrator's Guide* explains the integration between Service Desk and HP OpenView Operations for Windows and UNIX®. This guide covers the installation and configuration of the integration and explains how to perform the various tasks available with the integration.

The OpenView Operations Integration Administrator's Guide is available as a PDF file on the HP OpenView Service Desk 4.5 for Windows CD-ROM. The file name is `OVO_Integration_AG.pdf`.

- The *HP OpenView Service Desk: Migration Guide* provides a detailed overview of the migration from ITSM to Service Desk, to include an analysis of the differences in the two applications. Detailed instructions in this guide lead through the installation, configuration and other tasks required for a successful migration.

The Migration Guide is available as a PDF file on the HP OpenView Service Desk 4.5 for Windows CD-ROM. The file name is `Migration_Guide.pdf`.

- The *HP OpenView Service Desk: Web API Programmer's Guide* contains information that will help you create customized integrations with Service Desk using the Service Desk Web API. This API is particularly suited for developing Web applications.

The Web API Programmer's Guide is available as a PDF file on the HP OpenView Service Desk 4.5 for Windows CD-ROM. The file name is `Web_API_pg.pdf`.

- The *HP OpenView Service Desk: Data Dictionary* contains helpful information about the structure of the application.

The Data Dictionary is available as an HTML file on the HP OpenView Service Desk 4.5 for Windows CD-ROM. The file name is `Data_Dictionary.htm`.

- The *HP OpenView Service Desk 4.5 Computer Based Training (CBT)* CD-ROM is intended to assist you in learning about the functionality of HP OpenView Service Desk 4.5 from both a user and a system administrator perspective. The CD-ROM contains demonstration videos and accompanying texts that explain and show how to perform a wide variety of tasks within the application. The CBT also explains the basic concepts of the Service Desk application.

The *HP OpenView Service Desk 4.5 Computer Based Training (CBT)* CD-ROM will be shipped automatically with the regular Service Desk software. The CBT will be available for shipment shortly after the release of the Service Desk software.

- The online help is an extensive information system providing:
 - procedural information to help you perform tasks, whether you are a novice or an experienced user;
 - background and overview information to help you improve your understanding of the underlying concepts and structure of Service Desk;
 - information about error messages that may appear when working with Service Desk, together with information on solving these errors;
 - help on help to learn more about the online help.

The online help is automatically installed as part of the Service Desk application and can be invoked from within Service Desk. See the following section entitled “Using the Online Help” for more information.



Reading PDF Files

You can view and print the PDF files with Adobe® Acrobat® Reader. This software is included on the HP OpenView Service Desk 4.5 CD-ROM. For installation instructions, see the `readme.htm` file on the CD-ROM.

The latest version of Adobe Acrobat Reader is also freely available from Adobe’s Internet site at <http://www.adobe.com>.

Using the Online Help

You can invoke help from within Service Desk in the following ways:

- To get help for the window or dialog box you are working in, do one of the following:
 - Press **F1**.
 - Click the help toolbar button .
 - Choose **Help** from the **Help** menu.
 - Click the help command button  in a dialog box.

- To search for help on a specific subject using the table of contents or the index of the help system: choose Help Contents & Index from the Help menu.



When you are in the help viewer, you can find help on how to use the help system itself by clicking the Help toolbar button:




Service Desk also provides tooltips and “What’s This?” help for screen items like buttons, boxes, and menus.

A *tooltip* is a short description of a screen item. To view a tooltip, rest the mouse pointer on the screen item. The tooltip will appear at the position of the mouse pointer.

“*What’s This?*” help is a brief explanation of how to use a screen item. “What’s This?” help generally gives more information than tooltips. To view “What’s This?” help:

1. First activate the “What’s This?” mouse pointer in one of the following ways:
 - Press **Shift+F1**.
 - Click the “What’s This?” toolbar button .
 - Choose What ‘s This? from the Help menu.
 - In dialog boxes, click the question mark button  in the title bar.

The mouse pointer changes to a “What’s This?” mouse pointer .

2. Then click the screen item for which you want information. The “What’s This?” help information appears in a pop-up window.

To close the pop-up window, click anywhere on the screen or press any key on your keyboard.

Typographic Conventions

The table below illustrates the typographic conventions used in this guide.

Font	What the Font Represents	Example
<i>Italic</i>	References to book titles Emphasized text	See also the <i>HP OpenView Service Desk: Installation Guide</i> . <i>Do not delete</i> the System user.
Bold	First-time use of a term that is explained in the glossary	The service call is the basis for incident registration.
Courier	Menu names Menu commands Button names File names Computer-generated output, such as command lines and program listings	You can adjust the data view with the commands in the View menu. Choose Save from the menu. Click Add to open the Add Service Call dialog box. To start the installation, double-click <code>setup.htm</code> . If the system displays the text <pre>C:\>dir a:</pre> The device is not ready then check if the disk is placed in the disk drive.
Courier bold	User input: text that you must enter in a box or after a command line	If the service call must be solved within 30 minutes, enter 30.

Font	What the Font Represents	Example
<i>Courier italic</i>	Replaceable text: text that you must replace by the text that is appropriate for your situation	Go to the folder <i>X:\Setup</i> , where <i>X</i> is your CD-ROM drive.
Helvetica bold	Keyboard keys A plus sign (+) means you must press the first key (Ctrl in the example), hold it, and then press the second key (F1 in the example).	Press Ctrl+F1 .

We Welcome Your Comments!

Your comments and suggestions help us understand your needs, and better meet them. We are interested in what you think of this manual and invite you to alert us to problems or suggest improvements. You can submit your comments through the Internet, using the HP OpenView Documentation Comments Web site at the following URL:

http://ovweb.external.hp.com/lpe/comm_serv

If you encounter errors that impair your ability to use the product, please contact the HP Response Center or your support representative.

The latest versions of OpenView product manuals, including Service Desk manuals, are available on the HP OpenView Manuals Web site at the following URL:

http://ovweb.external.hp.com/lpe/doc_serv

Software patches and documentation updates that occur after a product release, will be available on the HP OpenView Software Patches Web site at the following URL:

<http://support.openview.hp.com/cpe/patches>

Requirements

HP OpenView Service Desk is currently being tested in different environments so that we can provide detailed information on the products required and supported with Service Desk 4.5.

Details on supported product versions cannot be supplied at this time. Supported product versions will be tested and certified for use with Service Desk 4.5. This information will be available with the general release of Service Desk 4.5.

For the purpose of beta testing, details on the requirements will be provided in the `readme.htm` file supplied with the beta software.

This section lists the minimum hardware requirements for each of the components of Service Desk 4.5. For software requirements, please refer to the *HP OpenView Service Desk: Supported Platforms List*. You can find this document on the HP OpenView Service Desk 4.5 CD-ROM, under the file name `\Doc\Supported_Platforms_List.htm`.

Service Desk Client on Windows 2000 or XP Professional

- Processor: Intel® Pentium® PII, 300 MHz
- Memory: 128 MB RAM
- Disk space: 55 MB
- Display: 800 x 600 resolution, 256 colors (1024 x 768 resolution, high color or higher recommended)
- Network connection: TCP/IP, 28.8 K (56 kB or higher recommended)

Service Desk Client on Windows NT4 Workstation and 98 SE

- Processor: Intel® Pentium® Pro, 200 MHz
- Memory: 128 MB RAM
- Disk space: 55 MB
- Display: 800 x 600 resolution, 256 colors (1024 x 768 resolution, high color or higher recommended).
- Network connection: TCP/IP, 28.8 K (56 kB or higher recommended)

Service Desk Client on Windows 2000 Terminal Services

The Service Desk client is supported for use with Windows Terminal Service. For information regarding on sizing please refer to the white paper created by Microsoft, located on the following Web site:
<http://www.microsoft.com/windows2000/techinfo/administration/terminal/tscaling.asp>

Service Desk Application Server on Windows 2000 Advanced Server or NT4 Server

- Processor: Intel Pentium PIII Xeon, 550 MHz
- Memory: 512 MB RAM
- Disk space: 80 MB
- Network connection: TCP/IP, 10 MB/s

Service Desk Application Server on HP-UX

- HP PA-RISC machine
- Disk space: 80 MB

Service Desk Application Server on Sun Solaris

- Sun Ultra SPARC machine
- Disk space: 80 MB

Service Desk Database Server

- Network connection: TCP/IP, 10 MB/s

Service Desk Service Pages Client

- Processor: Intel Pentium 90 MHz (166 MHz or higher recommended)
- Memory: 32 MB RAM
- Display: 800 x 600 resolution, 256-colors (1024 x 768 resolution, high color or higher recommended)
- Network connection: TCP/IP, 28.8 K (56 kB or higher recommended)

Service Desk Service Pages Web Server

- Memory: 128 MB RAM (256 MB or higher recommended)
- Disk space: 50 MB
- Network connection: TCP/IP, 10 MB/s

Service Desk Event Communicator

Service Desk Agent • Disk space: 5 MB

- Network connection: TCP/IP, 28.8 kB

Service Desk Service Event • Disk space: 5 MB

- Network connection: TCP/IP, 28.8 kB

1 **Installation**

The installation is straightforward. You only need to include three files in your Java class path.

Class files

Overview

There is a single file that contains all classes of the Web API:

- `web-api.jar`

This file contains all the generated classes, utility classes, and classes that manage the communication with the server that the API programmers can use in their own software.

Usage

You need to include this file in your class path to be able to use the Web API. This file is located in the API folder of the relevant CD-ROM within the Service Desk distribution pack.

2 **API Structure**

The Web API consists of classes written in pure Java (that enable programmers to access and manipulate Service Desk data. The structure has three levels of hierarchy: session, home and entity.

Structure Outline

Server- and Client-Side Components

The Web API consists of two parts, the server-side component and the client-side component. The server-side component runs on the Service Desk application server, in the same Java virtual machine process as the Service Desk server application. Its task is to handle incoming calls from the client-side component.

The client-side component runs in a separate virtual machine, in a pure Java environment. This could be the same process as where a web server runs. This component consists of the API classes that are to be used by a programmer. It is this component that will be explained in this document.

Client-Side Structure

The classes on the client side fall into three different levels of hierarchy:

The first level consists of the session classes. A programmer will be using the class `ApiSDSession` to start up a session and from then on use all the functionality of the API.

The second level consists of the home classes. Each different type of entity has its own home class. The names of these classes are `Api<Entity Name>Home`, for example `ApiChangeHome` or `ApiServicecallHome`. These classes are implementations of interfaces, called `IChangeHome`, `IServicecallHome` etc. Users of the API will always use these interfaces, not the classes that implement them.

These classes are used, amongst others, to:

- open individual records of the entity to which the home class belongs;
- create new records;
- find records that meet some search criteria.

The home classes use two kinds of utility classes, called `Api<Entity Name>Where` and `Api<Entity Name>Entitlement`. These classes are used to build where clauses for a find action, and give information about entitlement, respectively.

The third level consists of the entity classes. Each different type of entity has its own entity class. The names of these classes are Api<Entity Name>, for example ApiChange or ApiServicecall. These classes are implementations of interfaces, named IChange, IServicecall etc. Users of the API will always use these interfaces, not the classes that implement them. These classes contain the values of attributes of individual records, and methods to manipulate these values.

All these entity-specific classes follow the same hierarchy by which they are defined in the Object Model of Service Desk. This is done by inheritance of classes. You can see the structure of this inheritance in the diagrams on the following pages. In the top of the structure there is a slight difference for entities and homes. This is because some entities cannot exist independently, but are always contained in another entity. This type of entity is called an aggregated entity, and it extends the IApiAggregatedEntity interface. As an example IAssignment is used.

Figure 2-1 Entity structure

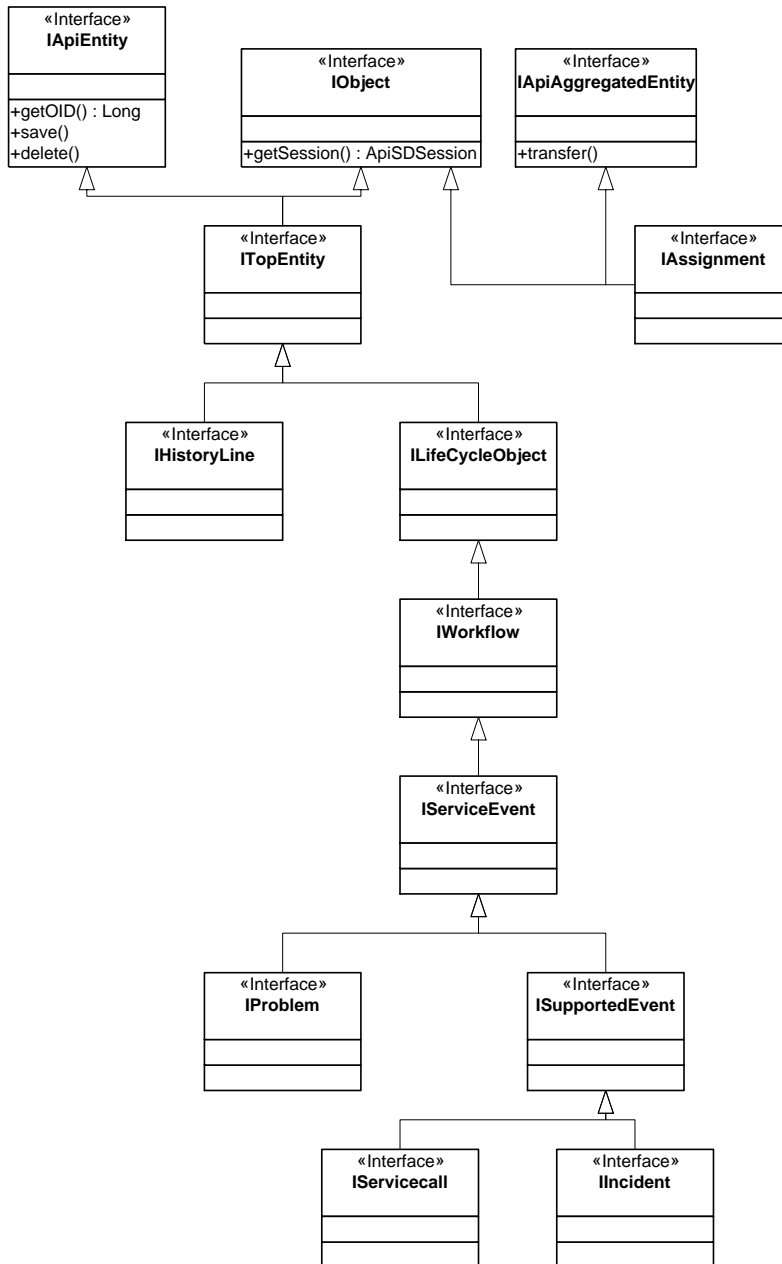
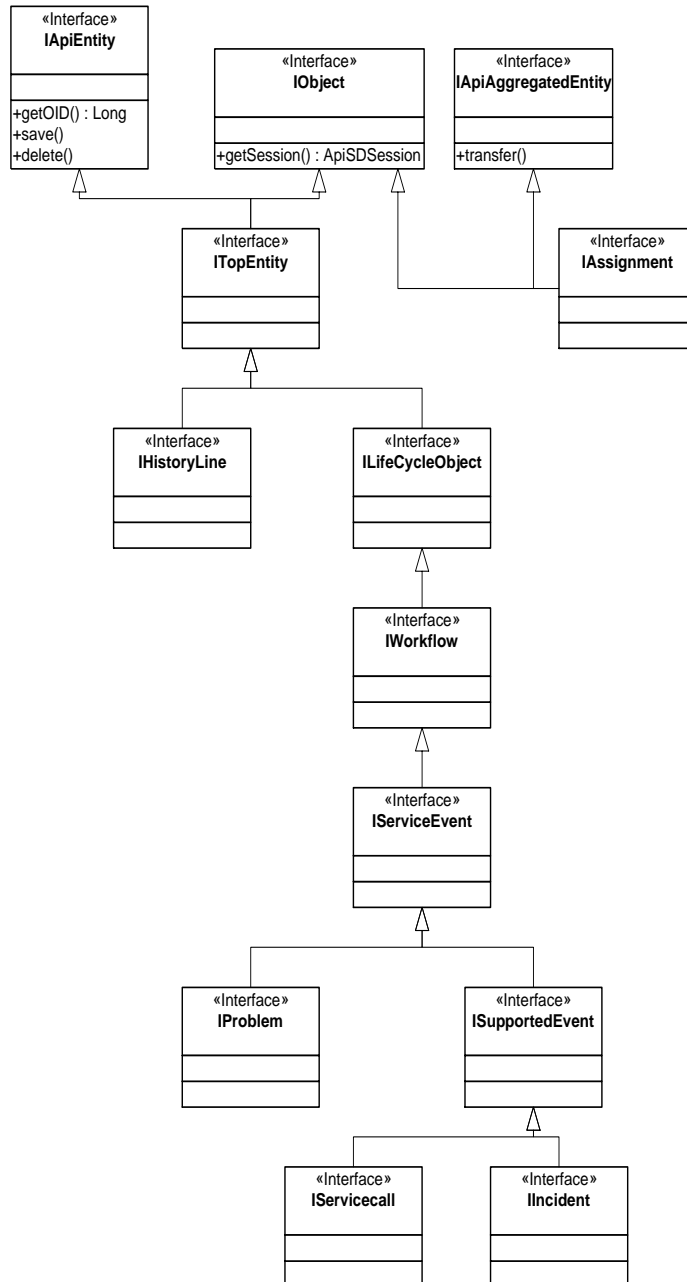


Figure 2-2 Home Structure



3 Overview of Available Methods

The interfaces that are available for the API user contain methods to retrieve and manipulate Service Desk data. This chapter describes the available methods.

Methods in Home Classes

There are two kinds of entities: abstract and non-abstract. Non-abstract entities are entities that are really used by users of Service Desk. Abstract entities only exist because non-abstract entities use them, by inheritance. Homes of non-abstract entities have more methods than homes of abstract entities.

First the methods for homes of non-abstract entities:

Open method

This method opens an existing record of this entity. The name of this method is `open<entity name>(value)`,

e.g. `openServicecall(value)`, or `openProblem(value)`. The passed value can be a `Long` (i.e. a long value wrapped in the object `Long`), meaning the Object Id of the record that you want to open, or a simple long, meaning the functional id of the record. Of course, this is only possible for entities that have a functional id. The return type is the entity interface type that corresponds with this entity. For example, the method `openServicecall` returns an `IServicecall`.

Open new method

This method returns an instance of the corresponding entity interface, representing a new record of this entity. The name of this method is always

`openNew<entity name>(value)`,

e.g. `openNewServicecall(value)`, or `openNewProblem(value)`. The passed value is optional. When given, it must be a `Long` value, representing the template that you want to use for filling in initial values.

Find methods

These methods return an array of instances of the corresponding entity interface, which meet search criteria. There are three variants:

- `findAll<entity name>` : use no criteria, just return all.

- `find<entity name>(where)` : use criteria from where. This argument is an instance of an entity-specific where class. This will be discussed later. If where is null, then this method is equivalent to `findAll`.
- `find<entity name>(Long view)`. The argument view is a Long value that represents the Object Id of the view, of which you want to use the search criteria (i.e. the filter defined by the view).

Search methods

These methods are nothing more than a simple version of the find method. The name is always

- `searchOn<attribute name>(value)`,
and the argument is the value of the attribute that you want to search on. For example: `searchOnDescription("Password forgotten")`. In case of Strings, the search is always case insensitive, but otherwise the value must be exactly the value you provided. If you also want the records that only contain this String, you must use the specific where interface.
- `void delete(Long id)`. This method deletes the record with the given object id.
- `Long getDefaultView()`. Get the Object Id of the default view of this entity.
- `get<Entity Name>Entitlement()`.
This method returns an instance of the interface `I<Entity Name>Entitlement`, which contains information about the entitlement for this entity.
- `create<Entity Name>Where()`.
This method returns an instance of the interface `I<Entity Name>Where`, that can be used to build a list of criterions with which you want to find values.

The following method is available to all homes, abstract or non-abstract:

- `String getLabel<attribute name>`. Returns the localized label of an attribute.

Methods in Entity Classes

Get and set value of attributes.

Each entity class has a get and set method for all attributes that are defined for that entity. The name of the get method is always

`get<attribute name>`,

and the set method is called

`set<attribute name>(value)`.

For example, `IWorkflow` has the methods `getDescription()` and `setDescription(String value)`. These methods are inherited by the classes that extend `IWorkflow`.

A more detailed discussion of get and set methods follows later.

Non-aggregated entities:

`void save()`. This method saves changes that are made, or inserts the record if it is new.

`void delete()`. This method deletes a record.

`Long getOID()`. Return the unique id of a record.

Aggregated entities:

* `void transfer()`. This method informs the entity to which an aggregated entity belongs, that there are changes in this aggregated entity, that need to be taken into account when the non-aggregated entity is saved. This method is the equivalent of `save()` for non-aggregated entities.

Utility Classes

Entitlement Classes

Each entity has its own entitlement class, called `Api<entity name>Entitlement`, e.g. `ApiServicecallEntitlement`. These classes are implementations of the interfaces `I<entity name>Entitlement`, e.g. `IServicecallEntitlement`. These interfaces have methods for all attributes of its entity, indicating whether these attributes may be modified and whether they are required.

Where Building Classes

Each entity has its own where building class, called `Api<entity name>Where`, e.g. `ApiServicecallWhere`. These classes are implementations of the interfaces `I<entity name>Where`, e.g. `IServicecallWhere`. You can get an instance of this interface by calling the method `create<entity name>Where()` on the home the entity. These interfaces have methods to add search criteria to the where. This where interface can be used as an argument for the find methods in the home of the entity.

All attributes of this entity have their own method to add a criterion, called:

`addCriterionOn<attribute name>(value)`,

The passed value is the value of the attribute that you want to search on.

For attributes that have a simple Java type, or attributes that reference another entity, that has an ordering attribute, there is the method:

`addCriterionOn<attribute name>Range(arg[])`

This way, you can specify the two values that the attribute value must be between. The argument of this method is an array, containing two values, namely the low boundary value, and the high boundary value.

Lastly, for String values there is the method:

addContainCriteriumOn<attribute name>(String value)

This simply says the result must contain the value.

NOTE

If you specify more than one criterion, the criteria are combined by a logical AND. In other words, the result of the search must meet all the criteria that are specified.

Getting and Setting Attribute Values

As stated earlier, each attribute has its own get and set method. The get method returns the value of the attribute, and the set method allows you to give the attribute a value. The Java type of this value follows from the method definition.

There are four categories of types:

- The first category is formed by the attributes that have basic Java types as values, such as String, Long, Boolean, etc.;
- The second category consists of attributes that are a reference to another entity. These attributes are called entity reference attributes;
- The third category consists of attributes that reference a set of entities, such as history lines. These attributes are called entity set reference attributes;
- The fourth category is actually a special kind of entity set reference, namely the n-m relation attributes.

These four categories are explained below.

Getting and Setting Attributes of Basic Types

An example of this kind is the attribute 'Description' in the abstract entity 'Workflow'. The get and set methods for this attribute are defined in the interface IWorkflow, and are called:

```
String getDescription()  
void setDescription(String value)
```

All entities that inherit from Workflow (like Service call, Problem, Incident) also automatically inherit these methods.

Another example is the attribute 'Concurrent user' in Account. This attribute says whether an account is for a concurrent user or not. The get and set methods for this attribute are defined in the interface IAccount, and are called:

```
Boolean getConcurrentUser()  
setConcurrentUser(Boolean value)
```

Getting and Setting Entity Reference Attributes

An example is the attribute 'Status' in the entity 'Problem'. The get and set methods are defined in the interface IProblem. The value of this attribute references another entity, namely the entity 'StatusProblem'; therefore the value of this attribute is of type IStatusProblem. So the methods are defined as:

```
IStatusProblem getStatus()  
void setStatus(IStatusProblem value)
```

Another example is the attribute 'ConfigurationItem' in the abstract entity ServiceEvent. The get and set methods are defined in the interface IServiceEvent. The value of this attribute references another entity, namely the entity 'ConfigurationItem'; therefore the value of this attribute is of type IConfigurationItem. So the methods are defined as:

```
IConfigurationItem getConfigurationItem()  
setConfigurationItem(IConfigurationItem value)
```

NOTE

For attributes that reference an aggregated entity, there is no set method. This is because an aggregated entity can never exist on its own, and therefore there is no need for a set method. For example if you want to set the assignment of an incident, you can do that as follows:

```
IAssignment assignment = incident.getAssignment() ;  
assignment.setAssigneePerson(assignPerson);  
assignment.transfer();
```

In this example incident is an instance of IIncident, and assignPerson is an instance of IPerson. The transfer method is used to notify the incident to which the assignment belongs, that there are changes in it.

Handling Entity Set Reference Attributes

An example is the attribute 'Historyline' in the entity 'Servicecall'. The value of this attribute is a reference to a set of records of the entity 'Historyline'. In the API, it is represented in the form of an array of the interface IHistoryLineServicecall. For this type of attribute, there cannot be a set method, instead there is an add method, that adds a history line to the service call. Adding a history line to an entity can only be done if that entity already exists.

So the methods look like this

```
IHistoryLineServicecall[] getHistoryLines()
void addHistoryLine(IHistoryLineServicecall value)
```

Handling n-m Relation Attributes

An n-m relation attribute is an attribute that references a set of records of an entity. This is used, for example, for the relation between persons and workgroups. A workgroup can contain more than one person, but a person can also be a member of more than one workgroup. There is one extra method for this kind of attributes namely the `unrelate` method. This method is used to break the relation between two entities, without removing the entities themselves. Also the `add` method is different. In this case, the entity that is added must be an existing one. The only thing that is added is the relation itself. So in the interface `IWorkgroup` there are the methods:

```
IPerson[] getMember()
addMember(IPerson value)
unrelateMember(IPerson value)
```

Another example is the parent-child relation structure between configuration items. Configuration items can have more than one child, and also more than one parent. So this relation is also an n-m relation. The following methods are defined in `IConfigurationItem` to handle these relations:

```
public IConfigurationItems[] getChildConfigurationItems()
public IConfigurationItems[] getParentConfigurationItems()
public void addChildConfigurationItem(IConfigurationItem value)
public void addParentConfigurationItem(IConfigurationItem value)
public void unrelateChildConfigurationItem(IConfigurationItem value)
public void unrelateParentConfigurationItem(IConfigurationItem value)
```

Java Comment for Attribute Methods

All methods with `get` or `set` attribute values have Java comment, explaining what the method is for. Actually, this comment is not made by hand, but generated automatically. The structure is always as follows. First a general description is given of what the method does. Then comment about the attribute is given. This comment about the attribute is copied from the 'what's this' help about the attribute. Not all attributes have a 'what's this' explanation though, in these cases, only the general comment is given.

Reverse Get Methods

There is one type of get method that has not been discussed. This kind of get method is a so-called 'get used by' method. It is not related to a specific attribute of the entity in which the get method is defined. It can be explained best by an example. Suppose you have an instance of the `IStatusServicecall` interface. This interface represents the status of a service call. Now you want to know what service calls have this status, in other words, by which service calls this status is used. In `IStatusServicecall` there is a method to do this:

```
IServicecall[] getServicecall_Status()
```

This method simply returns an array of instances of `IServicecall`, of which the field 'Status' references this `IStatusServicecall`.

Another example is the method:

```
IChange[] getChange_Manager()
```

in the interface `IPerson`. This method returns all changes of which the field 'Manager' references this person. In other words, with this method you can get all changes of which this person is the manager.

Web API Application

Service Desk 4.5 includes a user interface that enables you to select Service Desk items and fields to build a Web Api application. The advantage of using this tool is that you can specify which fields in a record are to be queried. This gives better performance as whole records do not need to be queried when you query on fields that are not part of the default set.

To use this Web Api selection tool:

1. Start a Service Desk client as Administrator.
2. From the **Tools** menu select **System** to open the Administrator's Console.
3. In the Administrator's Console expand the **Data** node, and select **Web Api Application**.
4. To create a new Web Api Application right-click in the right-hand window, and select **New Web Api Application** from the popup menu. Item and field selection is described in the on-line help.

The field sets you create using this tool are grouped under the name **Web Api Application** because each one you build is a set of configurations that you use by specifying the name in the method:

```
ApiSDSession.setApplicationSettings(WebApiApplication)
```

Where `WebApiApplication` is the configuration set you created in the Web Api Application interface. In this way you treat the configuration set as any other entity, and can search for a record in the standard way. For an example see "Example7.java" on page 42.

The Web API Javadoc

A Javadoc document for the Service Desk Web API is provided on the relevant CD-ROM within the Service Desk distribution pack. The Javadoc consists of a set of hyperlinked HTML files, generated from the API source code, that describe the classes, interfaces, and methods of the API.

You can find the Javadoc in the Doc folder on the relevant CD-ROM within the Service Desk distribution pack. The file name is Web API Javadoc.zip. Before you can use the Javadoc, you must extract all the files from the Web API Javadoc.zip file to a location of your choice. Make sure to select the Use folder names option for the extraction. During the extraction, a folder named html is created in the folder you specified as target location. To open the Javadoc, open the index.html file in the html folder.

Examples

A number of example classes are provided to demonstrate the use of the API. The examples are located in the `Api\examples` folder on the relevant CD-ROM within the Service Desk distribution pack.

The code should be self-explanatory; comments are added to guide you through the different steps.

The following examples are available:

Example1.java

Shows how to open a session, and obtain some information about the session and the account that opens it.

Example2.java

Shows how to open an existing service call, knowing its functional id. Also shows how to obtain information about that service call, and related entities.

Example3.java

Shows how to create a new problem, set some values, and save it.

Example4.java

Shows how to open an existing incident, change some values, and save it.

Example5.java

Shows how to make a where, containing criteria, and use it in a find.

Example6.java

Shows how to get a set of service calls, using the search criteria of a view.

Example7.java

Shows how to use the application specific settings that can be defined in the administrator console.

4 **Appendix**

This appendix notes some changes between Service Desk 4.0 and 4.5.

Changes from Service Desk 4.0 to 4.5

Because the Object Model of Service Desk has changed somewhat between 4.0 and 4.5, it is inevitable that the Web API has changed also. Most changes are new attributes or name changes of attributes. In some cases however, the structure of the relation between entities has changed also.

Most notably, in one case a pure n-m relation between 2 entities has become two 1-n relations. This is the case for the predecessor/successor relation between IWorkorders. This is because the relational entity IWorkorderPredecessor has new attributes, which stops it from being a pure relational entity.

This means the following methods have disappeared from IWorkorder:

- -addPredecessor(IWorkorder value)
- -IWorkorder[] getPredecessor()
- -unrelatePredecessor(IWorkorder value)
- -addSuccessor(IWorkorder value)
- -IWorkorder[] getSuccessor()
- -unrelateSuccessor(IWorkorder value)

The following methods are now available in the interface IWorkorder:

- -addPredecessor(IWorkorderPredecessor value)
- -IWorkorderPredecessor[] getPredecessor()
- -addSuccessor(IWorkorderPredecessor value)
- -IWorkorderPredecessor[] getSuccessor()

The following methods are now available in the interface IWorkorderPredecessor:

- -IWorkorder getPredecessor()
- -setPredecessor(IWorkorder value)

- -IWorkorder getSuccessor()
- -setSuccessor(IWorkorder value)

Another change is the relation between IConfigurationItem and IOrganization. This has become a pure n-m relation now, so the following methods are now available:

IConfigurationItem:

- -addCiOrg(IOrganization)
- -unrelateCiOrg(IOrganization)
- -IOrganization[] getCiOrgs()

IOrganization:

- -unrelateConfigurationItem(IConfigurationItem)
- -addConfigurationItem(IConfigurationItem)
- -IConfigurationItem[] getConfigurationItems()

The old methods to link IConfiguration and IOrganization have disappeared.

