Open Geographical Consortium
Java interfaces as OGC standards

Martin Desruisseaux
(Geomatys)
Werner Keil
(Creative Arts & Technologies)
Ingo Simonis
(Freelancer)
and others
What is the OGC?

• Not-for-profit
• International industry consortium
• Founded 1994, currently 340+ members
• Open Standards development by consensus process

OGC Mission

To lead in the development, promotion and harmonization of open spatial standards ...
OGC Program

Interoperability Program – a global, innovative hands-on rapid prototyping and testing program designed to accelerate interface development and validation and to bring interoperability to the market.

Specification Development Program
– Consensus standards process similar to other Industry consortia (World Wide Web Consortium, OMA, OASIS, JCP etc.).

Outreach and Community Adoption Program – education and training, encourage take up of OGC specifications, business development, communications programs
What is GeoAPI?

• OGC/ISO specifications as programmatic interfaces
• Analogous to JDBC, but for geospatial applications
Why GeoAPI

Isn't OGC Web Services sufficient?

• Similar to JDBC:
  – SQL existence doesn't mean that low-level API is not needed.
  – JDBC interfaces complete SQL, and they proven to be quite useful.
• Easily switch from one toolkit to another (demo)
• Mix components from different toolkits (demo)
• Reduce the learning curve
How GeoAPI is designed

• Sources are UML in OGC/ISO specifications
• Adapted to meet expectations of Java developers (departures documented in the GeoAPI specification)
Why Java interfaces derived from OGC UML

• Give to developers a model designed by OGC/ISO experts
  – Help to anticipate problems that developers may encounter only years later. Often the OGC/ISO experts have already debated such problems.

• Developers can implement only the interfaces they need
  – Nevertheless, the full set of interfaces is still useable as “hook” for future developments.

• Implementors can refer users to existing documentation
  – Less documentation effort for implementors
  – Model more likely to be familiar to users
/**
 * Abstract coordinate reference system, defined by a coordinate system and a datum.
 *
 * @since 2.0
 */

@UML(identifier="SC_CRS", specification=ISO_19111)
public interface CoordinateReferenceSystem extends ReferenceSystem {

/**
 * Returns the coordinate system.
 */

@UML(identifier="usesCS", specification=ISO_19111, obligation=MANDATORY)
CoordinateSystem getCoordinateSystem();

/**
 * Returns the datum.
 */

@UML(identifier="usesDatum", specification=ISO_19111, obligation=MANDATORY)
Datum getDatum();
}
Where interfaces stand

- Between OGC/ISO specifications and implementations
- Java language for now, but other languages are possible
- Many implementations for the same set of interfaces
Implementation flexibility

• Comparable to JDBC
  – PostgreSQL, Oracle or MS-Access databases don't have to be implemented in a “JDBC way”

• Concepts can be merged for simplicity
  – Demonstrated by Proj.4 wrapper and examples
History

Java interfaces for Coordinate Transformation Services

First draft of Geographic Objects specification (GO-1)

Creation of GeoAPI on SourceForge as an attempt to join effort between GeoTools and OpenJump

2001

GO-1 and GeoAPI join effort: Creation of a GeoAPI WG at OGC

2002

GeoAPI WG dissolved. Creation of GeoAPI 3.0 SWG

2003

GO-1 final version; GeoAPI 2.0 release

2005

GeoAPI 3.0.0 release as an OGC standard

2008

2011

WG: Working Group

SWG: Standard Working Group
GeoAPI Implementation Specification

GeoAPI 3.0 Implementation Standard

1) Overview
2) Downloads
3) Related Links
4) Related News

1) Overview

The GeoAPI Implementation Standard defines, through the GeoAPI library, a Java language application programming interface (API) including a set of types and methods which can be used for the manipulation of geographic information structured following the specifications adopted by the Technical Committee 211 of the International Organization for Standardization (ISO) and by the Open Geospatial Consortium (OGC). This standard standardizes the informatics contract between the client code which manipulates normalized data structures of geographic information based on the published API and the library code able both to instantiate and operate on these data structures according to the rules required by the published API and by the ISO and OGC standards.

2) Downloads

<table>
<thead>
<tr>
<th>Version</th>
<th>Document Title (click to download)</th>
<th>Document #</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.0</td>
<td>GeoAPI 3.0 Implementation Standard</td>
<td>09-083r3</td>
<td>IS</td>
</tr>
</tbody>
</table>

http://www.opengeospatial.org/standards/geoapi
OGC standard working group

- Technical discussions on the SourceForge mailing list
- Procedural (votes, etc.) discussions on the OGC mailing list
  - Any OGC member can join
  - Decisions are done according the OGC rules
  - Only OGC staff can deploy to Maven Central
**Evolution**

---

**getEditionDate**

```java
@UML(identifier="editionDate", 
    obligation=OPTIONAL, 
    specification=ISO_19115)
Date getEditionDate()
```

Date of the edition, or null if none.

**Warning:** The return type of this method may change in GeoAPI 3.1 release. It may be replaced by a type matching more closely either ISO 19108 (*Temporal Schema*) or ISO 19103.

**Returns:**

The edition date, or null if none.

---

**All impacted API:**

- `Metadata.getDateStamp()`
- `Citation.getEditionDate()`
- `CitationDate.getDate()`
- `Element.getDates()`
- `Event.getTime()`
- `ProcessStep.getDate()`
- `RequestedDate.getLatestAcceptableDate()`
- `RequestedDate.getRequestedDateOfCollection()`
- `Requirement.getExpiryDate()`
- `MaintenanceInformation.getDateOfNextUpdate()`
- `StandardOrderProcess.getPlannedAvailableDateTime()`
- `Usage.getUsageDate()`
- `Datum.getRealizationEpoch()`
- `TemporalDatum.getRealizationEpoch()`
- `TemporalDatum.getOrigin()`
- `DatumFactory.createTemporalDatum(Map, Date)`
What is Sensor Web?

- Geologic Sensor Web
- Ecosystem Monitoring Sensor Web
- Weather Forecasting Sensor Web

- Users
  - Science
  - Decision support
  - Disaster response
  - Government/P policy
  - Industry
  - Public

- Command/Control/Coordination Data Processing
- Weather Radar Network
- Seismic/GPS Sensor Network
- Modeling
Scientific View: Sensor Web

- Applications
- Sensor Web
- Sensor Network
- Sensor System
- Sensor

Communication
OGC: Sensor Web
Sensor Web Intent

- Quickly **discover sensors and sensor data** (secure or public) that can meet my needs – location, observables, quality, ability to task

- **Obtain sensor information** in a standard encoding that is understandable by me and my software

- Readily **access sensor observations** in a common manner, and in a form specific to my needs
Sensor Web Intent II

- Task sensors, when possible, to meet my specific needs
- Subscribe to and receive alerts when a sensor measures a particular phenomenon
Sensor Web Vision I

- Sensors will be web accessible
- Sensors and sensor data will be discoverable
- Sensors will be self-describing to humans and software (using a standard encoding)
- Most sensor observations will be easily accessible in real time over the web
Sensor Web Vision II

- Standardized web services will exist for accessing sensor information and sensor observations

- Sensor systems will be capable of real-time mining of observations to find phenomena of immediate interest

- Sensor systems will be capable of issuing alerts based on observations, as well as be able to respond to alerts issued by other sensors
Sensor Web Vision III

- Software will be capable of on-demand geolocation and processing of observations from a newly-discovered sensor without *a priori* knowledge of that sensor system.

- Sensors, simulations, and models will be capable of being configured and tasked through standard, common web interfaces.

- Sensors and sensor nets will be able to act on their own (i.e. be autonomous).
Sensor Web Vision

- Wireless sensor networks (WSN)
- Image Processing Service
- Notification Threshold Service
- Fusion Processing Service
- Storage Service
- Authentication Service
- Registry Service
- Web Mapping Client
- End Users Application
- Sensors Mission Planning System

OGC®
Sensor Web: Building Blocks

• Quickly **discover sensors and sensor data** (secure or public) that can meet my needs – location, observables, quality, ability to task
Sensor Web: Building Blocks

- **Obtain sensor information** in a standard encoding that is understandable by me and my software
Sensor Web: Building Blocks

Services
- Observation
- Tasking
- Alerting
- Analysis

Registries & Dictionaries
- Sensor Description Language
- Phenomenon Description Language
What is KML?

- KML is a file format used to display geographic data in an Earth browser, such as
  - Google Earth,
  - Google Maps
  - Google Maps for mobile
  - etc.

- Who uses KML
  - Casual Users
  - Scientists
    - E.g. mapping Earthquakes
  - Non-Profits
    - Humanitarian missions like UN in Dafur
  - Students and Educators
www.opengeospatial.org