# OpenOffice.org Software Development Kit (SDK)

## For own solutions and products based on OpenOffice.org and Java

by
Jürgen Schmidt
jsc@openoffice.org

# Agenda

- About the speaker
- Overview of the OpenOffice.org Software Development Kit (SDK)
  - What is it good for
- The UNO component model
  - Introduction, Concepts, Key features
- The OpenOffice.org API
- Usage Scenarios
- Examples
- Outlook

# About the speaker

- Jürgen Schmidt
- Technical Lead Software Engineering
  - have been working for StarOffice/Sun more than 5 years
  - involved in UNO development since the beginning in 1997
- OpenOffice.org/StarOffice Software Development Kit

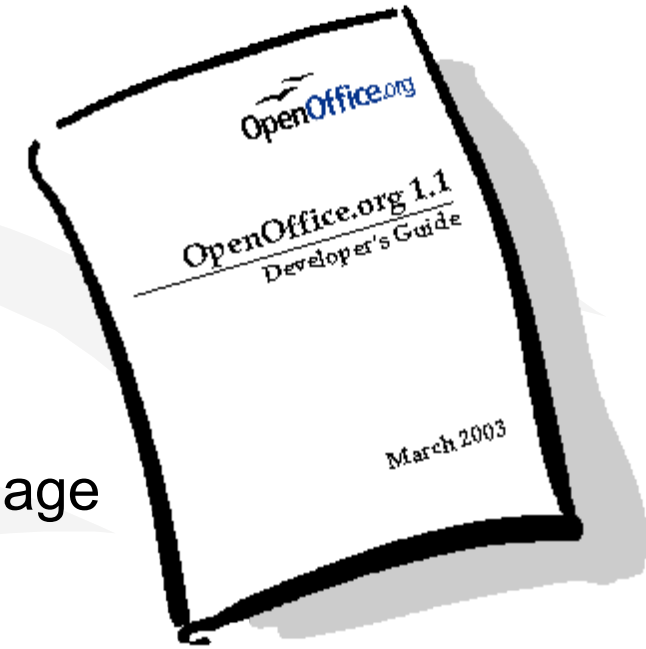
*UNO supporter and fan*

The essential extension for all who want to program, extend or control OpenOffice.org

- What is the SDK?
  - Add-on to an existing office installation
  - Documentation
    - Developer's Guide
    - IDL reference
    - Java/C++ UNO runtime and helper classes/functions
  - Tools and Libraries
  - Examples
    - Java, C++, OpenOffice.org Basic and OLE

# SDK Documentation

## Developer's Guide




- ~ 900 pages
  - a growing document
- covers the whole API
  - each chapter provides at least one example
  - target language Java (with exception of language specific chapters)
- PDF and HTML version
  - HTML version provides cross references into the IDL reference for easy navigation

# IDL Reference

- generated documentation based on the IDL sources
  - cross references in the Developer's Guide
    - where the type is described mainly
    - where the type is referenced
  - references of type usage
    - as return type
    - as parameter
    - as member
    - ...
  - generated with autodoc
  - Javadoc like

Overview Module Use Devguide Index

METHODS' SUMMARY METHODS' DETAILS ATTRIBUTES' SUMMARY ATTRIBUTES' DETAILS

:: com :: sun :: star :: text ::

interface XText

**Base Hierarchy**

::com::sun::star::uno::XInterface
 |
 +-XTextRange
    |
    +-XSimpleText
       |
       +-XText

**Description**

extends a XSimpleText by the capability of inserting XTextContent s.

**Developers Guide**

7.3.1 Text Documents - Working with Text Documents - Word Processing - Editing Text - Text Contents Other Than Strings

7.3.1 Text Documents - Working with Text Documents - Word Processing - Editing Text

# SDK Tools and Libraries

- Deployment tool (pkgchk)
- IDL compiler (idlc)
- “interface” generators (cppumaker, javamaker)
- Type library tools (regmerge, regview, regcompare, regcomp)
- Documentation tool (autodoc)
- UNO component loader (uno)
  - tool for loading components and provide an instance of them as a named object
- OfficeBean
  - Jar file and native library
  - since OpenOffice.org 1.1 part of the office installation
- UNO protocol library (prot_uno_uno)
  - is a bridge: binary uno ⇔ binary uno
  - core feature

# SDK Tools and Libraries

## Deployment tool

- easy deployment of
  - components
  - configuration
  - OpenOffice.org Basic libraries
- part of the office installation
  - deploying of final extensions without SDK
- currently no live deployment

# SDK Examples

- Developer's Guide
  - Java, C++, OpenOffice.org Basic
- Java
  - component examples
  - “remote” control
  - several API areas (text, drawing, spreadsheet, ...)
  - Lotus Notes Access
- C++
  - core UNO
  - Filter example
    - used for the Developer's Guide
- OpenOffice.org Basic
- OLE
  - ActiveX control, VB script, Delphi

# The UNO component model

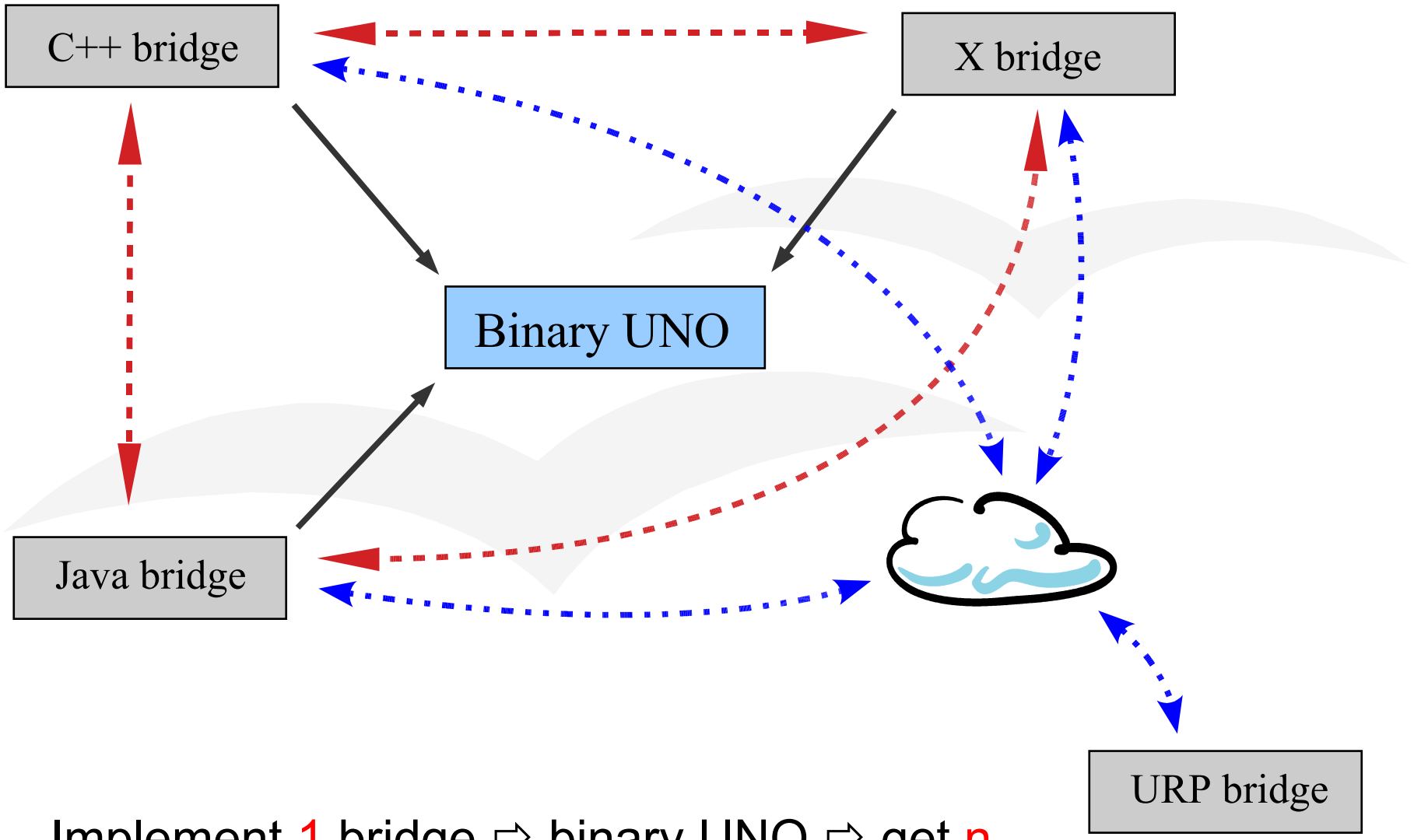
## Introduction

- UNO ⇨ Universal Network Objects
- Why UNO?
  - started in June 1997, at this time no sufficient component technology was available
  - providing a component technology
    - implementations are exchangeable
    - more flexibility
    - hide implementation details
    - new external API
- Now in the third generation

# UNO Concepts

- Language Bindings
  - bridge, runtime, component loader, ...
- Abstract definition of interfaces and services
  - use of an Interface Definition Language (IDL)
- Mediation between UNO environments
- Factory concept

# UNO Concepts




Implement 1 bridge ⇨ binary UNO ⇨ get n

# UNO Features

- Language independent
  - C, C++ (various compiler)
  - Java
  - OpenOffice.org Basic
  - OLE
- Seamless remote interoperation
  - remote calls are transparent
  - preserves thread and object identity
  - supports asynchronous calls
- No overhead in case of co-location
- Supports exception
- Multi threaded

# UNO Features

- Has a security concept, which is derived from the Java security model
- Uses Unicode for strings
- No code generation
- Basically independent from OpenOffice.org
- Remote protocols exchangeable
  - currently used: URP (UNO Remote Protocol)
- UNO Url for getting remote objects
  - e.g. uno:socket, host=localhost,port=8100;urp;MyObject

# The OpenOffice.org API

- Design goal
  - One API for all
    - macros
    - use components
    - exchange/modify components
    - extend functionality by new components
- Programming against specifications
- UNO objects
  - service based
    - UNO objects should implement at least one service
  - Instantiation
    - by a factory using a service name
    - context dependent
      implicitly by accessing subobjects, return value or out parameter

## Supported IDL types

- service
  - abstract object specification
  - implementation independent
  - we have to kind of services
    - services which can be instantiated directly
    - abstract base services
- interface
  - only methods
    - in/out/inout parameter
    - Exceptions
  - no data
  - no implementation
  - independent of any programming language
- modules

## Supported IDL types

- struct
  - data containers of different types
  - support inheritance
  - easy be transferable into other UNO environments
- exception
  - support inheritance
- enum
  - similar to a “C” enum type
- const
- constant
  - group of constants
  - used to categories functional dependent consts

## Common Design Patterns

- Factory
  - global and document centric
- PropertySet, PropertyAccess, ...
- Collection/Containers
- Enumerators/Iterators
- X...Supplier
- Events
- Exceptions for error handling

Module structure

- UNO base
- application independent
- miscellaneous components
  - e.g. Configuration Manager, Universal Content Broker
- environment integration framework
- application domain specific
- office components

# Usage Scenarios

- Macro programming
  - automated tasks
  - forms
- “remote” control
  - document conversion
  - report generation
- Extensions ⇨ Components
  - specialized components (e.g. Calc Add-ins)
  - more complex extension
    - own menu items and/or dialogs
    - complex functionality
- Embedded in own GUI applications
  - OfficeBean

## Calc Add-in

- mandatory service
  - com.sun.star.sheet.AddIn
- mandatory interfaces
  - com.sun.star.lang.XServiceInfo
  - com.sun.star.lang.XTypeProvider
- own Add-in service
  - com.sun.star.sheet.addin.ExampleAddIn
    - with own Add-in interface XExampleAddIn
- Further requirements/constraints
  - display names and descriptions for the functions and parameters
  - Add-in functions have restricted set of possible return and parameter types

# Define your own Add-in

```
module com {
	module sun {
		module star {
			module addin {
				interface XExampleAddIn : com::sun::star::uno::XInterface {
					/// Sample function that just increments a value.
					long getIncremented( [in] long nValue );

					/// Sample function that returns a volatile result.
					com::sun::star::sheet::XVolatileResult getCounter( [in] string aName );
				};

				service ExampleAddIn {
					/// specify the mandatory base service which have to be implemented
					service com::sun::star::sheet::Addin;

					/// our own Add-in interface with our Add-in functions
					interface XExampleAddIn;
				};
			};
		};
	};
};
```

# Implementation class

```
public class ExampleAddIn
{
	static public class _ExampleAddIn extends com.sun.star.lib.uno.helper.WeakBase implements
			com.sun.star.sheet.addin.XExampleAddIn, com.sun.star.sheet.XAddIn,
			com.sun.star.lang.XServiceName, com.sun.star.lang.XServiceInfo
	{
		....
		private static final String[] aFunctionNames = { "getIncremented", "getCounter" };
		private static final String[] aDisplayFunctionNames = { "Increment", "Counter" };
		....
		//  XExampleAddIn
		public int getIncremented( int nValue ) {
			return nValue + 1;
		}
		....
		//  XAddIn
		public String getProgrammaticFuntionName( String aDisplayName ) {
			for ( int i = 0; i < aFunctionNames.length; i++ )
				if ( aDisplayName.equals( aDisplayFunctionNames[ i ] ) )
					return aFunctionNames[i];
			return "";
		}
		....
	}
}
```

# Implementation class

```
public class ExampleAddIn
{
	static public class _ExampleAddIn extends com.sun.star.lib.uno.helper.WeakBase implements
			com.sun.star.sheet.addin.XExampleAddIn, com.sun.star.sheet.XAddIn,
			com.sun.star.lang.XServiceName, com.sun.star.lang.XServiceInfo
	{
		static private final String aExampleService = "com.sun.star.sheet.addin.ExampleAddIn";
		....
	}

	/// required component function to get a factory for objects of this service implementations
	public static com.sun.star.lang.XSingleServiceFactory __getServiceFactory(String implName,
			com.sun.star.lang.XMultiServiceFactory multiFactory,
			com.sun.star.registry.XRegistryKey regKey) {
		com.sun.star.lang.XSingleServiceFactory xSingleServiceFactory = null;
		if ( implName.equals(_ExampleAddIn.aImplName) )
			xSingleServiceFactory = com.sun.star.comp.loader.FactoryHelper.getServiceFactory(
				_ExampleAddIn.class, _ExampleAddIn.aExampleService, multiFactory, regKey);
		return xSingleServiceFactory;
	}

	/// required component function to get information about the implementation, used for registration
	public static boolean __writeRegistryServiceInfo(com.sun.star.registry.XRegistryKey regKey) {
		....
	}
}
```

# Outlook

- IDE integration (wizards)
- specialized OfficeBeans (e.g. WriterBean, CalcBean)
- Scripting Framework
- ANT build scripts for Java examples
  (popular build tool for Java, IDE independent)
- more examples
  - real life examples,
  - Developer's Guide examples in C++/OpenOffice.org Basic
  - improved documentation of the source code
- improved developer documentation
  - extended and improved Developer's Guide
  - improved reference documentation (IDL, Java, C++)
- maybe simplified wrapper APIs
  (depends on user demand)

# Links & questions

- OpenOffice.org http://www.openoffice.org
- API project http://api.openoffice.org
- UDK project http://udk.openoffice.org

# Questions?