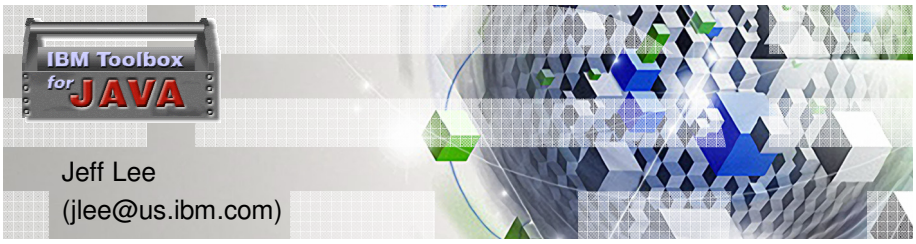
Session Number: 403552

# Introducing...
# The IBM Toolbox for Java™




Jeff Lee
(jlee@us.ibm.com)

Power your planet.

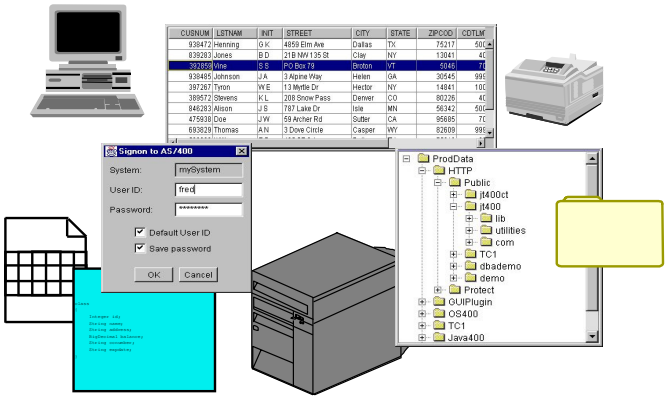




## IBM Toolbox for Java™

*What is the Toolbox/JTOpen?*

A set of Java classes and utilities which provide access to IBM i® data and resources

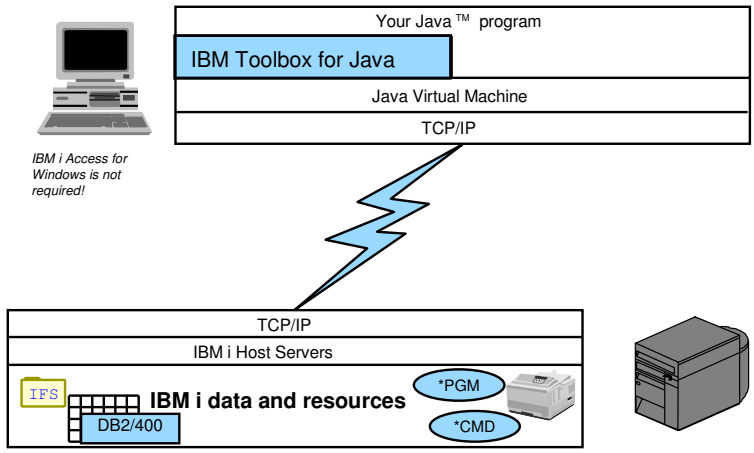





Power your planet.

# IBM Toolbox for Java

*The big picture - Client/Server*



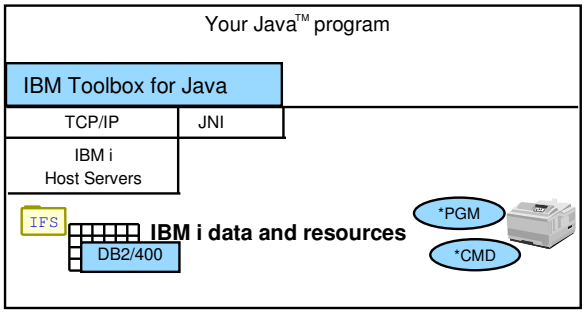






# IBM Toolbox for Java

*The big picture - Toolbox and data on same IBM i*

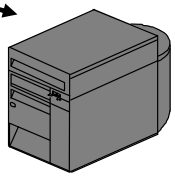


# IBM Toolbox for Java

*Using the Toolbox in client/server applications*

- Toolbox installed on client
- Java application runs on client
- IBM i Access for Windows is *not* required
- *The same Java application runs on any client with a Java-compatible JVM!*

TCP/IP



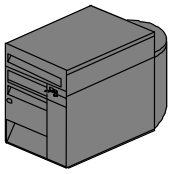
- Server running IBM i
- Uses existing IBM i host servers
- IBM i Java Virtual Machine (JVM) is *not* required on the server






# IBM Toolbox for Java

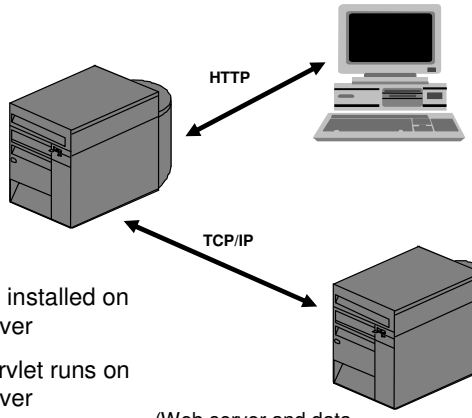
Using the Toolbox in *server applications*



- Toolbox installed on IBM i
- Java application runs on IBM i
- Use Toolbox instead of JNI
- "Local" sockets used to communicate between Toolbox and host servers
- Direct API calls used in some cases to bypass the host servers
- IBM i with Java Virtual Machine (JVM) is required

# IBM Toolbox for Java

*Using the Toolbox in Internet-based servlets*




- Web browser displays HTML
- No Java on the client

- Server running IBM i
- Uses existing IBM i host servers
- IBM i Java Virtual Machine (JVM) is *not* required

- Toolbox installed on web server
- Java servlet runs on web server
- Java-enabled web server is required

(Web server and data server can be the same machine)






# IBM Toolbox for Java

**Supported platforms:**

- **IBM i, Linux, Windows, AIX**, Solaris, Mozilla Firefox, Microsoft Internet Explorer

**Requires Java 1.1.8 or greater, and supports Java 2**

**Divided into packages:**

- com.ibm.as400.**access** - APIs for accessing IBM i data and resources
- com.ibm.as400.**resource** - Framework for accessing list-based data *(deprecated)*
- com.ibm.as400.**vaccess** - GUI components *(deprecated)*
- com.ibm.as400.**data** - Program call markup language
- com.ibm.as400.**ui**.* - Graphical Toolbox
- com.ibm.as400.**util**.* - HTML, XSL-FO and Servlet components
- com.ibm.as400.**micro**.* - APIs for wireless devices
- **utilities** - utility classes such as JarMaker, JPing, RunJavaApplication, AboutToolbox

# IBM Toolbox for Java

## *Packaging*

Licensed program 5722-JC1 (V5R4) or 5761-JC1 (V6R1) (no additional charge)

Downloadable from **JTOpen website** (no charge)

Ships with IBM i – in directory /QIBM/ProdData/HTTP/Public/jt400/lib/

Jar files:

- **jt400**.jar - Base function + GUI components
- **jt400Native**.jar - Base function only, intended for use on IBM i JVM
- **jt400Proxy**.jar - Proxy support, subset of jt400.jar
- **jt400Servlet**.jar - HTML, XSL-FO, and Servlet components
- **jt400Micro**.jar - Wireless support
- **uitools**.jar, **jui400**.jar, **util400**.jar - Graphical Toolbox
- **tes**.jar - System Debugger

Use the **JarMaker** utility to reduce the size of jt400.jar or any other jar file






# JTOpen (Open Source)

All of the primary Toolbox packages are open source
*http://sourceforge.net/projects/jt400*

- Part of IBM's open source development community
- Use source code as a debug tool
- Submit new function under the IBM Public License (IPL)
- Modify source for your use
- Submit problem reports and bug fixes

Two delivery modes of the Toolbox:

- Licensed program
  - Supported by IBM
  - Fixes are delivered as PTFs
- Open source version
  - Supported by IBM
  - New releases are available as free Web downloads
  - New functions and fixes available here first

# IBM Toolbox for Java

*Popular Toolbox Functions*

- Database access via **JDBC**
- Database access via a **record-level I/O** and **DDS** interface
- **Command Call**
- **Program Call** via both Java code and XML
- Data Queues / User Spaces / Data Areas
- Access files in IBM i **Integrated File System**
- Access **Print** object (spooled files, printers, queues, ...)
- Access other IBM i objects (**Jobs, Users, System Values**, etc.)
- Built-in automatic **data conversion**
- **HTML** / **Servlet** wrappers
- **Wireless** APIs
- XML-based **GUI Builder**
- Many components are Java Beans






# IBM Toolbox for Java

*IBM i products that are built on the Toolbox*

- IBM i Navigator and Management Central
- IBM i Access for Web
- IBM i Connect (B2B)
- IBM Host On Demand
- *plus many more...*

# IBM Toolbox for Java

*Access Classes: Low-Level Java APIs to Access Data*

- User Authentication and Identification
- Command Call
- Connection Pools
- Clustered Hashtables
- Data Area
- Data Description
- Data Conversion
- Data Queues
- Environment Variables
- FTP
- IFS
- JDBC
- Jobs
- Messages
- NetServer
- Print
- Permissions
- Program Call
- Record-level Database Access
- Save File
- System Status
- System Values
- Users and Groups
- User Space






# Infrastructure

*"The **AS400** object"*

Represents a **connection** to the IBM i

Provides a **sign-on GUI**

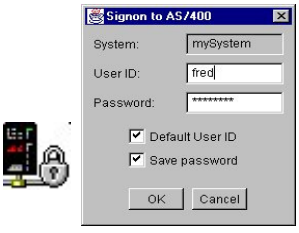
- Password caching available
- Change password GUI when appropriate

Controls conversations with server jobs

- Multiple users and multiple conversations
- Implicit and explicit connections

Provides Secure Sockets Layer (**SSL**) communication

- Encryption and server authentication




```
AS400 sys  = new AS400();
AS400 sys2 = new AS400("mySystem");
AS400 sys3 = new AS400("mySystem",
             "myUID", "myPWD");

CommandCall cc = new CommandCall(sys);
```

*Most Toolbox classes use the AS400 object*






# JDBC

*The Java standard for database access*

Write Java programs in terms of standard JDBC interfaces, then plug in *any* JDBC driver - to work with *any* database!

- Java gives you platform independence, JDBC gives you database independence

java.sql package in Java Developers Kit

SQL is used extensively

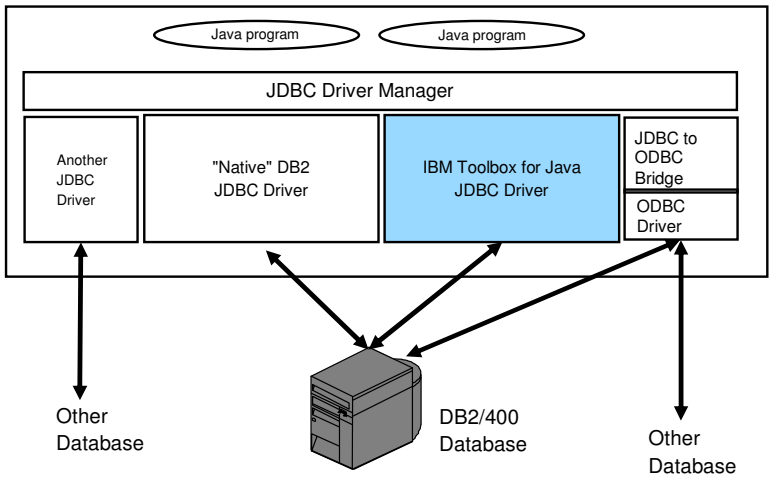
- Based on X/Open SQL Call Level Interface

Also supports:

- Database definitions, manipulations, and queries
- Stored procedures
- Catalog methods
- Transactions (commit, rollback, isolation levels, distributed)

# JDBC

*The Java standard for SQL database access*









# JDBC

*IBM i JDBC driver choices*

Toolbox JDBC driver (*com.ibm.as400.access.AS400JDBCDriver*)

- Communicates with the database using TCP/IP
- Great for:
  - client/server applications
  - applets
  - servlets, where the web server and data are not on the same IBM i

JDBC 4.0 support in JTOpen – requires Java 6.0

DB2/400 JDBC driver (*com.ibm.db2.jdbc.app.DB2Driver*)

- Communicates with the database using direct CLI calls
- Great for:
  - server applications
  - servlets, where the web server and data on same IBM i

```
DriverManager.registerDriver(...);
Connection c = DriverManager.getConnection(...);
Statement select = c.createStatement();
ResultSet rs = select.executeQuery("SELECT * FROM ...");
while (rs.next())
  System.out.println(rs.getString(column));
```

# Record-level database access

*Fast access to IBM i database files*

Provides access to database files:

- Access records sequentially, by record number, or by key
- Physical and logical file members are described by a RecordFormat
- Support for locking
- Support for transactions
- Familiar paradigm for RPG programmers
- Limited System/36 SSP file capability, too!

```
AS400 system = new AS400("mySystem");
SequentialFile file = new SequentialFile(system, "/QSYS.LIB/MYLIB.LIB/MYFILE.FILE");
file.setRecordFormat(...);
file.open(...);
Record r = file.readNext();
```





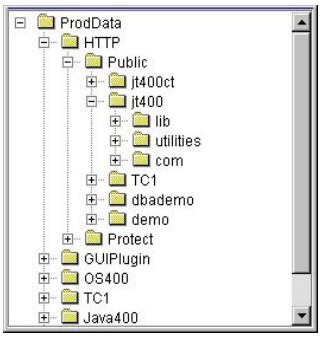

# Integrated file system

*File input, output, and more*

Implements standard Java input/output (*java.io*) classes:

- Read and write data sequentially or via random access
- Create, delete, and rename files and directories
- List the contents of a directory

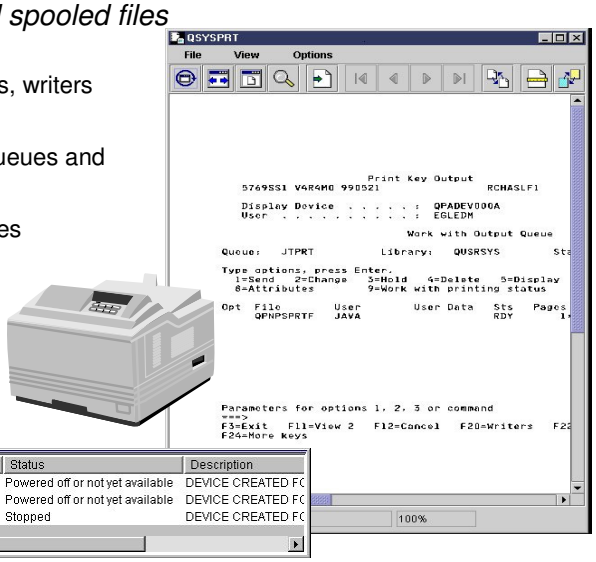
```
AS400 system = new AS400();

IFSFileOutputStream s = new
        IFSFileOutputStream(system, "/a.a");
byte[ ] data = new byte[n];
s.write(data);

IFSTextFileOutputStream s2 = new
        IFSTextFileOutputStream(system, "/b.b", 37);
s2.write("Hi Mom in EBCDIC");
```



# Network print

*Access print objects and spooled files*

- List printers, output queues, writers and spooled files
- Manage printers, output queues and writers
- Read and write spooled files
- Spooled file viewer





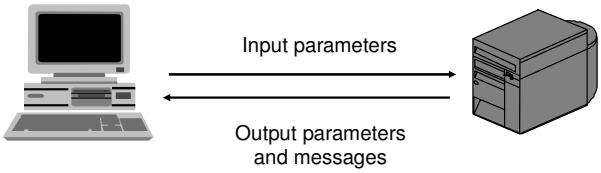




# Command call and program call

*Make use of legacy code and system APIs*

```
AS400 system = new AS400();
CommandCall cc = new CommandCall(system);
cc.run("CRTLIB NEWLIB");
```



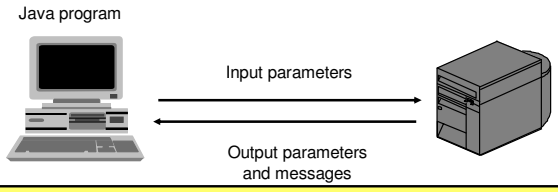

```
AS400 system = new AS400();
ProgramParameter[ ] parmList = new ProgramParameter[n];
parmList[0] = new ProgramParameter(data);
...
ProgramCall pc = new ProgramCall(system,
                    "/QSYS.LIB/MYLIB.LIB/MYPGM.PGM", parmList);
pc.run();
```

# Program Call Markup Language (PCML)

*Describe program calls using XML*

Automates program call parameter and structure definition

Simplifies data description and conversion




```
<pcml>
<program name="rtvsts" path="/QSYS.lib/MYLIB.lib/RTVSTS.pgm">
<data name="custid" usage="input" type="int">
<data name="status" usage="output" type="char" length="20">
</program>
</pcml>
```





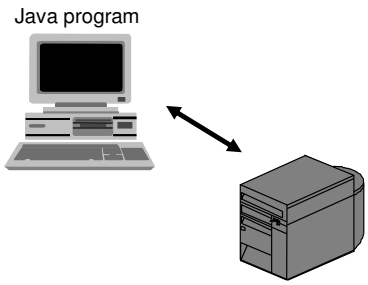

# Interprogram Communication

*Data Queues, Data Areas, User Spaces, ...*

Transfer data between programs using:

- Data areas
- Data queues (keyed or sequential)
- User spaces

Partner can be Java program, traditional IBM i program, on IBM i or another client




```
AS400 system = new AS400();
UserSpace us = new UserSpace(system,"/QSYS.LIB/MYLIB.LIB/MYDATA.USRSPC");
byte[ ] data = new byte[1024];
us.read(data, 0);
```

# RFML (Record Format Markup Language)

Very similar to **PCML** (Program Call Markup Language)

While PCML is designed specifically for Program Parameters,
RFML is generally useful for **parsing and composing**:

- Physical file records
- Data queue entries
- User spaces
- Data buffers

Specify record formats using XML; get/set field values

Segregate the data layout from the program logic






# RFML vs. FieldDescription

*Example: Composing a customer record*

**Using RFML:**

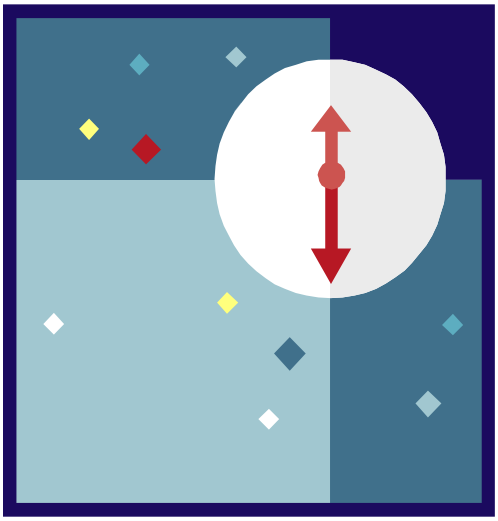
```
import com.ibm.as400.data.RecordFormatDocument;

RecordFormatDocument rfmlDoc =
                    new RecordFormatDocument("customer");
```

( In a separate file named "customer.rfml": )

```
<rfml version="4.0" ccsid="37">
  <recordformat name="cusrec">
    <data name="cusnum" type="int"   length="2" precision="16"/>
    <data name="lstnam" type="char"  length="8"/>
    <data name="baldue" type="zoned" length="6" precision="2"/>
  </recordformat>
</rfml>
```

**Without RFML:**

```
import com.ibm.as400.access.AS400Text;
import com.ibm.as400.access.AS400UnsignedBin2;
import com.ibm.as400.access.AS400ZonedDecimal;
import com.ibm.as400.access.BinaryFieldDescription;
import com.ibm.as400.access.CharacterFieldDescription;
import com.ibm.as400.access.RecordFormat;
import com.ibm.as400.access.ZonedDecimalFieldDescription;

RecordFormat recFmt = new RecordFormat("cusrec");

AS400UnsignedBin2 conv1 = new AS400UnsignedBin2();
BinaryFieldDescription desc1 = new BinaryFieldDescription(conv1, "cusnum");
recFmt.addFieldDescription(desc1);

AS400Text conv2 = new AS400Text(8, 37);
CharacterFieldDescription desc2 = new CharacterFieldDescription(conv2, "lstnam");
recFmt.addFieldDescription(desc2);

AS400ZonedDecimal conv3 = new AS400ZonedDecimal(6, 2);
ZonedDecimalFieldDescription desc3 = new ZonedDecimalFieldDescription(conv3,
"baldue");
recFmt.addFieldDescription(desc3);
```



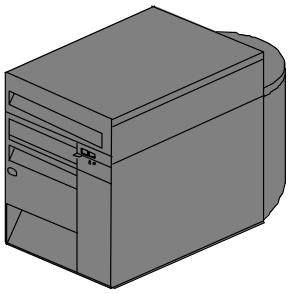





# Server Objects

*Jobs, Users, System Values, ...*

- List IBM i jobs
- List IBM i users and groups
- Display and change system values
- Manage message queues
- Manage user permissions to objects



```
AS400 system = new AS400();
SystemValue sv = new SystemValue(system, "QDATE");
System.out.println(sv.getValue());
```

# Data description and conversion

*Converts between Java data and IBM i data*

| *Java data type* | | *IBM i data type* |
|---|---|---|
| Object[ ] | ⟷ | Array |
| short | ⟷ | 2 byte binary |
| int | | 2 byte unsigned binary |
| Int | ⟷ | 4 byte binary |
| long | | 4 byte unsigned binary |
| long | | 8 byte binary |
| byte[ ] | ⟷ | Byte array |
| float | ⟷ | 4 byte floating point |
| double | | 8 byte floating point |
| BigDecimal | ⟷ | Packed decimal |
| BigDecimal | | Zoned decimal |
| Object[ ] | ⟷ | Structure |
| String | ⟷ | Text |

Handles all code page, byte order, and data conversion issues






# Data description and conversion

*Record formats*

Access data in the record by **field name**

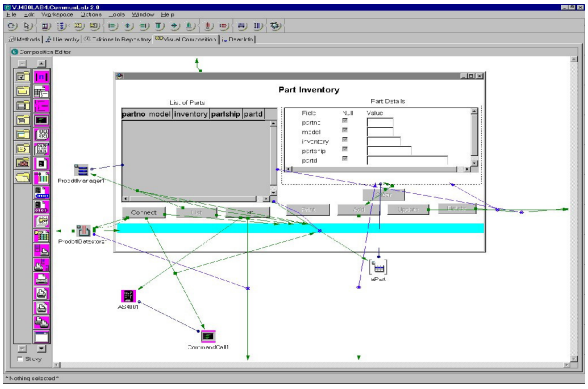
**Convert data automatically** for:
- Program calls
- Data queues
- Record-level database access; IFS text file access

```
BinaryFieldDescription customerNumber = new BinaryFieldDescription(new AS400Bin4(),
                                                    "CUSTOMER_NUMBER");
CharacterFieldDescription customerName = new CharacterFieldDescription
                                        (new AS400Text(20, system), "CUSTOMER_NAME");
RecordFormat recordFormat = new RecordFormat();
recordFormat.addFieldDescription(customerNumber);
recordFormat.addFieldDescription(customerName);
Record data = recordFormat.getNewRecord(dataQueue.read().getData());
Integer I = (INTEGER) data.getField("CUSTOMER_NUMBER");
String name = (String) data.getField("CUSTOMER_NAME");
```

# IBM Toolbox for Java

*Visual development environments*

Most Toolbox public classes are *Java Beans*. With visual development tools like WDSC (WebSphere Developer Studio client), no coding necessary!









# IBM Toolbox for Java

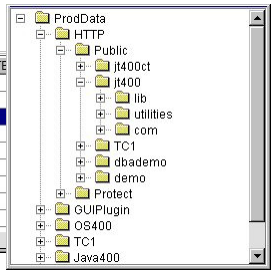
*Graphical user interface components*

A set of Java **GUI components** which present IBM i data and resources

Usable where a Java **Swing** GUI component (JComponent) is allowed

Toolbox requires Swing 1.1 via either the 1.1.x add-on, or Java 2.

| CUSNUM | LSTNAM | INIT | STREET | CITY | STATE |
|---|---|---|---|---|---|
| 938472 | Henning | G K | 4859 Elm Ave | Dallas | TX |
| 839283 | Jones | B D | 21B NW 135 St | Clay | NY |
| 392859 | Vine | S S | PO Box 79 | Broton | VT |
| 938485 | Johnson | J A | 3 Alpine Way | Helen | GA |
| 397267 | Tyron | W E | 13 Myrtle Dr | Hector | NY |
| 389572 | Stevens | K L | 208 Snow Pass | Denver | CO |
| 846283 | Alison | J S | 787 Lake Dr | Isle | MN |
| 475938 | Doe | J W | 59 Archer Rd | Sutter | CA |
| 693829 | Thomas | A N | 3 Dove Circle | Casper | WY |



# HTML and Servlet classes

*Web components create **tables** and **forms***

Provides **access** to database files:
- Access database file with Record Level Access or SQL via JDBC
- Includes Meta Data

Provides classes to **display data**:
- Display data in tables or forms
- Toolbox provides converters that will produce HTML tables or forms based on the row data

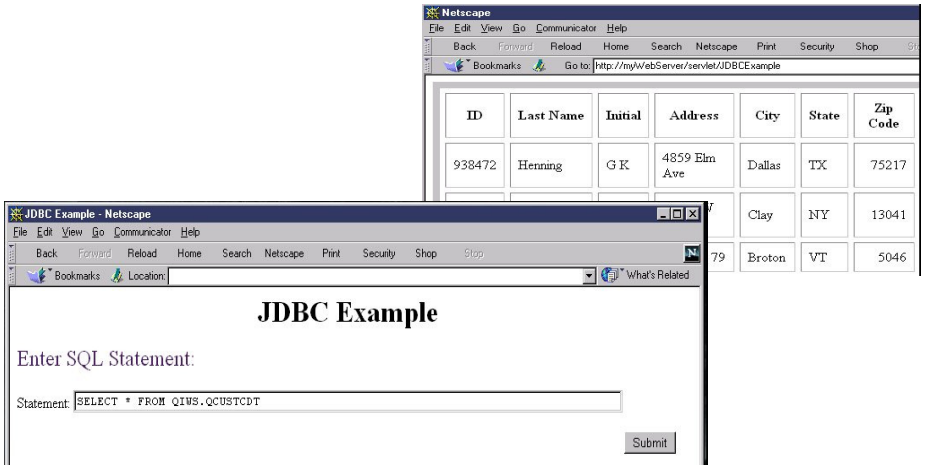
```
HTMLTableConverter converter = new HTMLTableConverter();

ResultSet resultSet = statement.getResultSet();
SQLResultSetRowData rowdata = new SQLResultSetRowData(resultSet);

String[ ] html = converter.convert(rowdata);
out.println(html[0]);
```






# HTML and Servlet classes

*Web components create tables and forms*



# HTML and Servlet classes

*Web components create tree hierarchy*

Provides classes to display the **Integrated File System**:

- **Display contents** of the Integrated File System
- Toolbox provides classes to create and display a customized and **traversable** tree

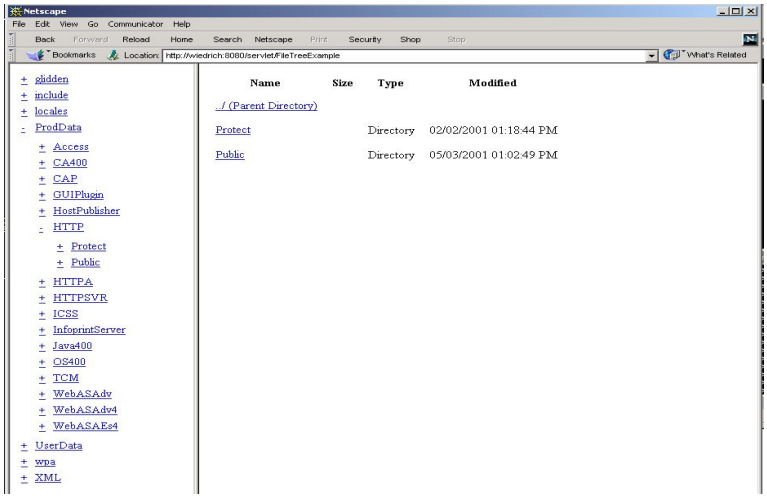
```
HTMLTree tree = new HTMLTree(HTTPrequest)

IFSJavaFile root = new IFSJavaFile(systemObject, "/QIBM");

DirFilter filter = new DirFilter();

File[] dirList = root.listFiles(filter);

for (int i=0; i<dirList.length; i++)
{
   FileTreeElement node = new FileTreeElement(dirList[i]);
   tree.addElement(node);
}
```

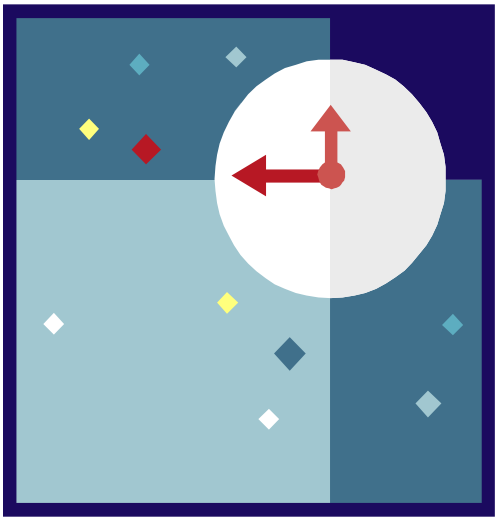





# HTML and Servlet classes

*Web components create **tree hierarchy***










# Toolbox Micro Edition

*Terms*

***What is ToolboxME?***

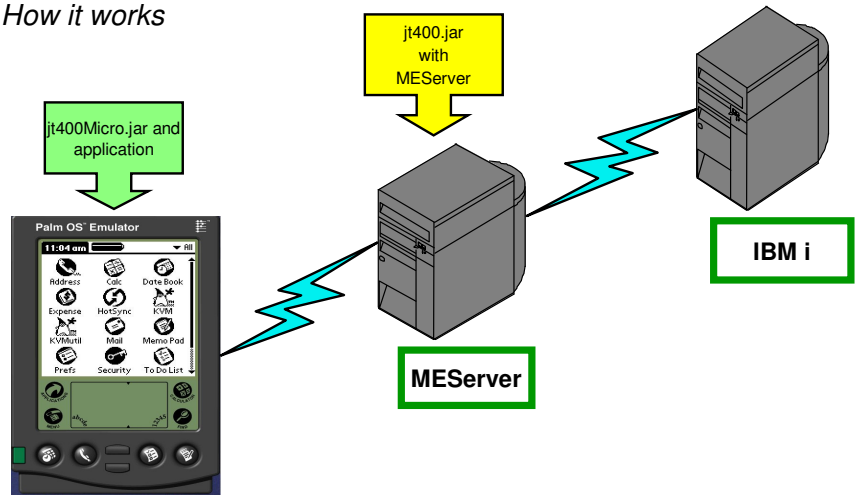
A subset of the Toolbox for Java classes which will provide access to IBM i data and resources from a **Tier 0 device**

***What is a Tier 0 device?***

- **Tier 3**: the IBM i server
- **Tier 2**: the application or web server
- **Tier 1**: the client desktop or laptop
- **Tier 0**: refers to the next level down. Tier 0 devices are predominantly thought of as web-enabled **cell phones** and personal digital assistants (**PDAs**).

# Toolbox Micro Edition

*How it works*









# Toolbox Micro Edition

*Supported Components*

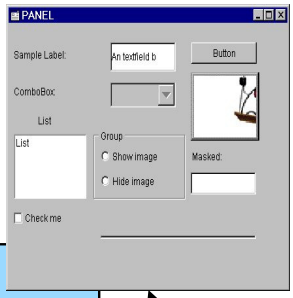
- AS400
- Command Call
- Program Call via PCML
- Data Queues
- JdbcMe

# Graphical Toolbox

*Describe GUI panels using **XML***

Panel Definition Markup Language (**PDML**)
- Simplifies GUI panel definition and layout

Resource script (RC) converter
- Converts Windows GUIs to Java




```
<pdml>
<panel name="order_entry" path="/QSYS.lib/MYLIB.lib/RTVSTS.pgm">
<title>Order Entry</title>
<button name="Ok" disabled="no">
   <title>Ok</title>
   <location>125,100</location>
   <size>100,26</size>
   <action>COMMIT</action>
</button>
</pdml>
```

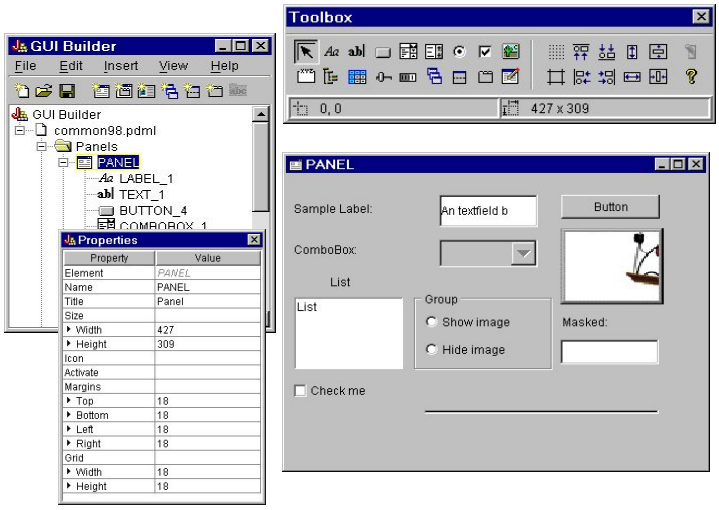





# Graphical Toolbox

*GUI builder*

- **WYSIWYG** panel definition
- Generates PDML code
- Now supports JavaHelp™



# Some other components

*The list keeps growing!*

- NetServer
- JarMaker
- IBM i Messages
- Message Files
- System Status
- Proxy Support
- Save File
- Report Writer
- Servlets
- System Properties

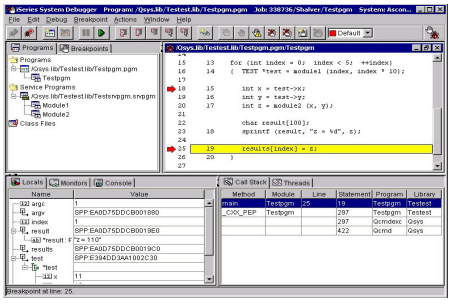
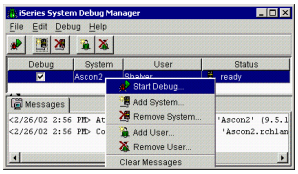








# System Debugger and Debug Manager

- Supports **all ILE languages**: C, C++, RPG, Java, COBOL, CL
- Point and click **breakpoint** manipulation in source code
- Automatic **variable evaluation** with mouse and **local variable display**
- Program **call stack** and **thread** display

- Requires JDK1.3 and **tes.jar**, jt400.jar, and jhall.jar

- Invoke with following: `java utilities.DebugMgr` *or*
  `java utilities.Debug -s system -u user`

# New in JTOpen as of V6R1

***Now available at www.ibm.com/systems/i/software/toolbox***

Other classes and features added in V6R1:

- JDBC enhancements including **generated key support**
- **JDBC 4.0** support (download from JTOpen)
- JDBC **performance** improvements
- AS400JDBCManagedConnectionPoolDataSource
- FileAttributes
- HistoryLog
- ObjectReferences
- UDFS (user defined file system)

Plus:

- CL command documentation generator
- Kerberos authentication is now supported through the use of JGSS
- Performance improvements in list processing (users, jobs, etc.)
- Unicode-enabled CL commands






# What's new since V6R1

- **"JC1" LPP eliminated as of IBM i 7.1**
  - Integrated into **SS1** (product ID 5770-SS1) **Option 3**
  - Same JAR files available in **same IFS directories** as in prior releases
- **New classes added**
  - Package `com.ibm.as400.access`
    - AS400JDBCArray, AS400JDBCArrayResultSet
    - ErrorCodeParameter
    - ObjectLockListEntry
    - UserObjectsOwnedList, UserObjectsOwnedListEntry
  - Package `com.ibm.as400.security.auth`
    - ProfileTokenProvider
    - DefaultProfileTokenProvider

# What's new since V6R1 - continued

- **Significantly *enhanced* classes**
  - Package `com.ibm.as400.access`
    - Many of the JDBC classes
    - CommandCall and ProgramCall - new thread-safety behavior, new methods
    - IFSFile, IFSJavaFile - new methods
    - AS400 - new system properties, new methods
    - AS400ConnectionPool
    - Trace
    - DataArea
    - SpooledFile
- For complete details, refer to the IBM i Information Center
  - http://publib.boulder.ibm.com/infocenter/iseries/v7r1m0/topic/rzahh/page1.htm






# Top 5 Good Things About the Toolbox

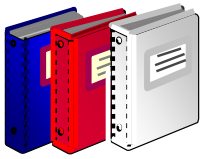
1. It's free, no strings attached.
2. Fully supported by IBM Service.
   - User forum on Web is monitored daily by IBM developers.
3. Lets any Java app, anywhere on your LAN,
   - Access and exploit your IBM i resources.
4. Thoroughly documented on the Web.
5. In use by IBM and customers since V4R2 (1998).
   - Used under-the-covers in many other IBM products.

# That's it!






# References



*Where can I get more information?*

www.ibm.com/systems/i/software/toolbox

- Toolbox for Java: News, downloads, FAQs, articles, COMMON labs

http://sourceforge.net/projects/jt400

- JTOpen - open source, bug reporting, feature requests

www.ibm.com/systems/support/i/forums

- IBM i Technical Forums - including IBM Toolbox for Java/JTOpen Forum

*IBM Toolbox for Java Programmers Guide*

- Shipped with the IBM Toolbox for Java
- Contains overview, full API documentation (javadoc), and code examples
- Available in the IBM i Information Center
  - http://publib.boulder.ibm.com/infocenter/iseries/v7r1m0/topic/rzahh/page1.htm

# Questions






**Please fill in the following information on your evaluation sheet:**

Session Title: **Introducing the IBM Toolbox for Java**

Session ID: **403552**

Agenda Key: **33SM**

Speaker: **Jeff Lee**

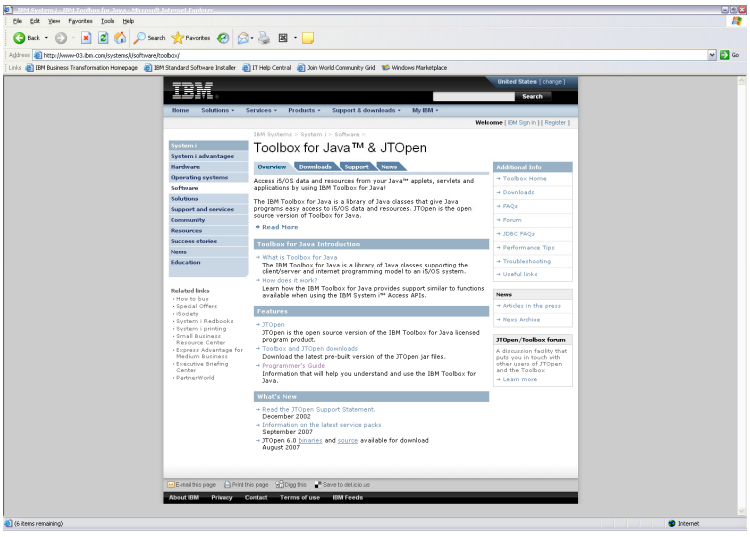
**Please fill in evaluation sheets and place in the bag**

# Backup Slides

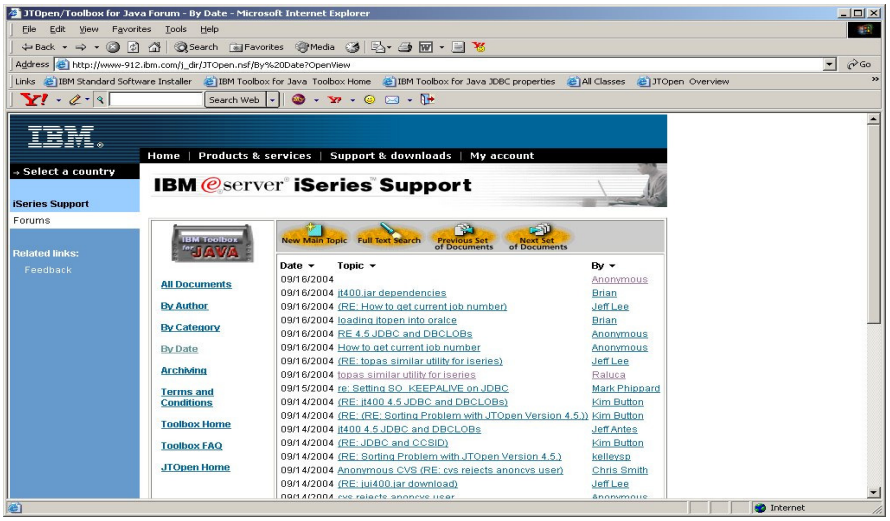





# IBM Toolbox for Java home page



# IBM Toolbox for Java/JTOpen Forum

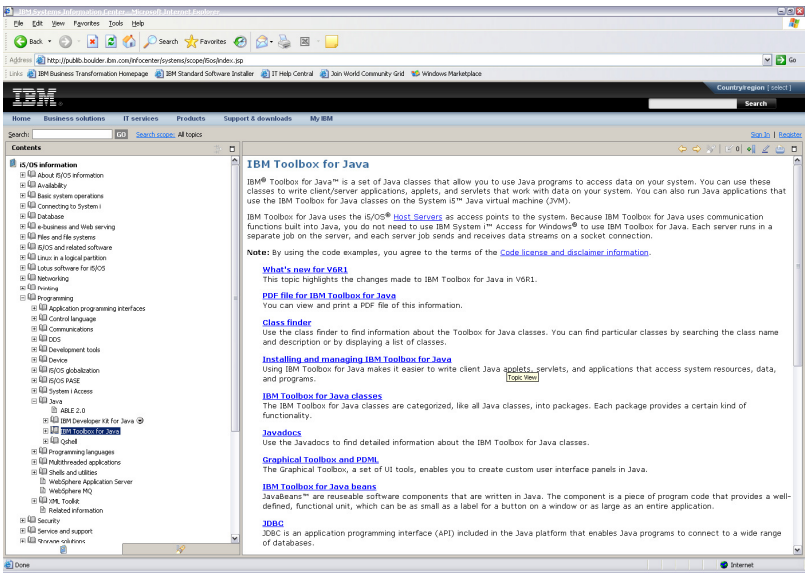




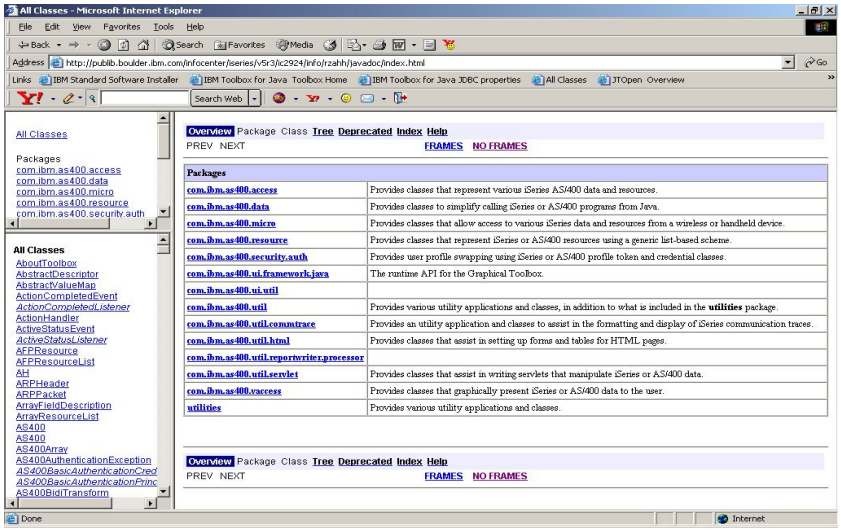




# Toolbox Programmer's Guide



# Javadoc









# JDBC Driver Types

- **Type 1 Driver - JDBC-ODBC bridge**
- **Type 2 Driver - Native-API Driver specification**
- **Type 3 Driver - Network-Protocol Driver**
- **Type 4 Driver - Native-Protocol Driver**
  - **The Toolbox JDBC driver is this type.**
- References:
  - http://java.sun.com/products/jdbc/driverdesc.html
  - http://en.wikipedia.org/wiki/JDBC_driver

# What's different in IBM i 7.1

## JDKs and JVMs

- The **LPP** for "IBM Developer Kit for Java" is **unchanged**: **5761-JV1**
  - Same LPP number as in IBM i 6.1
- "**Classic**" JDK is **not available** in IBM i 7.1
  - Replaced by "IBM Technology for Java" (code name: "J9")
  - The **no-longer-supported JV1 Options** that had "Classic" JVMs:
    - JV1 Options **6**, **7** and **10**
- **New Java Group PTF** number for IBM i 7.1
  - **SF99572** (versus SF99562 for IBM i 6.1)

For complete details, refer to the IBM i Information Center
- http://publib.boulder.ibm.com/infocenter/iseries/v7r1m0/topic/rzaha/rzahawhatsnew.htm






# What's different in IBM i 7.1 - *continued*

## "PASE for i" – Changes for improved security

*"IBM Portable Application Solutions Environment for i"*
- *Provides an **AIX-like execution environment** on IBM i.*
- *The "new" IBM i JVMs **require** a PASE environment.*

- PASE now **enforces stack execution disable** protection.
- Default behavior of PASE programs has changed.
  - **Instructions run from memory areas** (stack & heap) of a process are **blocked**.
  - JIT-generated code is created in memory areas.
    - If call `JNI_CreateJavaVM()`: **Must mark the program** as needing to **allow program execution** from memory areas.

For complete details, refer to the IBM i Information Center
- http://publib.boulder.ibm.com/infocenter/iseries/v7r1m0/topic/rzalf/rzalfwhatsnew.htm

# Trademarks and Disclaimers








# Special notices

# Special notices (cont.)



Revised February 9, 2010