EDT 0.8.1 Stakeholder Meeting

July 13, 2012

Agenda

- Meeting Goal --- Discuss the design and implementation new features to ensure they meet stakeholder needs

- Plans are subject to change (mostly based on your input)

- Feel free to ask questions via the phone or chat to All in the Web conference window

- Press *6 to mute/unmute your phone. Please mute your phone unless you are asking a question.

Today's Topics

▸ EDT Project Status – Lisa Lasher

▸ What's New in 0.8.1 M1 / M2

   - System Library support – Zhi Zhu, Fan Song, XiaoBin Chen

   - Language Enhancements – Justin Spadea

▸ What's Coming in M3 – Brian Svihovec

▸ What's Coming beyond M3 – Brian Svihovec

▸ EDT Community and Open Development – Theresa Ramsey

EGL Development Tools

EDT Project Status

Where We've Been

- EDT 0.7 Release – Dec 2
  - ▶ Initial incubator release
  - ▶ EGL model
  - ▶ Generator framework
  - ▶ IDE
  - ▶ Debugger
  - ▶ Deployment
  - ▶ Basic EGL language

- EDT 0.8 Release – April 13
  - ▶ Generator and debugger extensibility
  - ▶ Optimization of generated code
  - ▶ Integration with IBM I
  - ▶ JNDI support
  - ▶ IDE usability and productivity enhancements

Where We're At

- EDT 0.8.1 Schedule
  - ▶ M1 / M2 driver published June 11
  - ▶ M3 driver targeted to publish by June 29
  - ▶ Testing / fixing through July 20
  - ▶ 0.8.1 Release July 27
- For more on EDT 0.8.1 plans, see http://wiki.eclipse.org/EDT:0.8.1_Planning

Where We're Going

- EDT 0.8.2 – Fall 2012
  - ▶ Incremental enhancements in:
    - IDE usability / productivity
    - Extensibility
    - Language
- EDT 0.8.x ? TBD

- EDT 0.9
  - ▶ APIs are complete and frozen
  - ▶ Most (all?) functionality is complete
- EDT 1.0
  - ▶ Production-level functionality and quality

What's New In 0.8.1 M1 / M2

System Library Support - EGLARs

EGLAR Overview

- What is an EGL ARchive (EGLAR)?

  - An EGLAR file is a zip file that contains the compiled, intermediate representation (IR) version of one or more EGL parts. For more information about EGLAR files, see Introduction to EGLAR files and binary projects

- Role of EGLAR

  - An EGLAR is similar to a JAR file in Java

  - EDT 0.8.1 only supports system EGLAR files, which are provided with EDT. User defined EGLAR files cannot be created or consumed at this time.

EGL Path Changes

- The EGL System parts are now included on the EGL Path, instead of being implicitly defined by the compiler.

- A new container entry is added to .eglpath file that represents EGL system parts:

  ▸ `<eglpathentry kind="con" path="org.eclipse.edt.compiler.EDT_RUNTIME"/>`

  ▸ The entry is automatically added for newly created project in EDT 0.8.1.
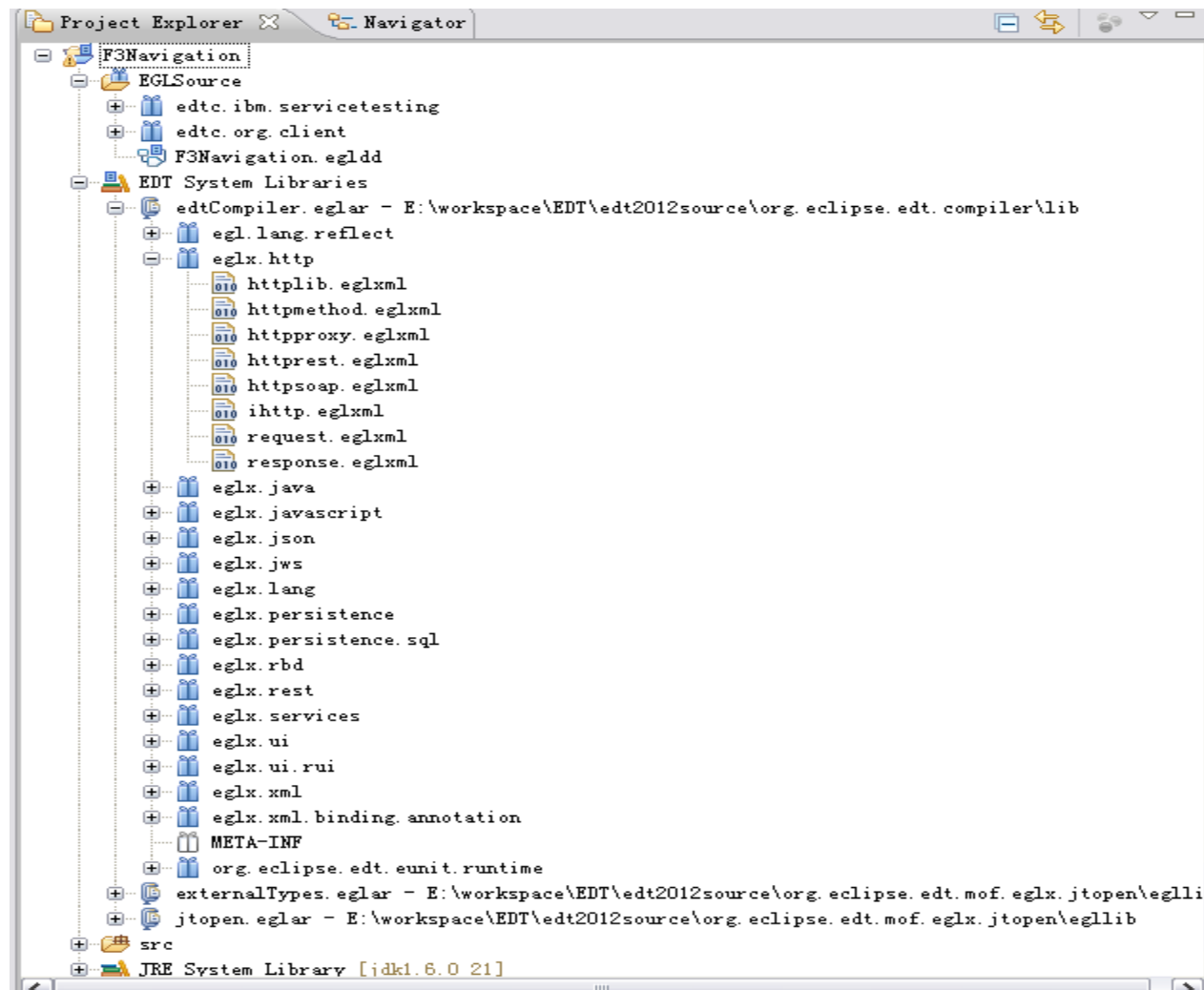
Project Explorer View Update

- EGL system parts are shown automatically in the Project Explorer View
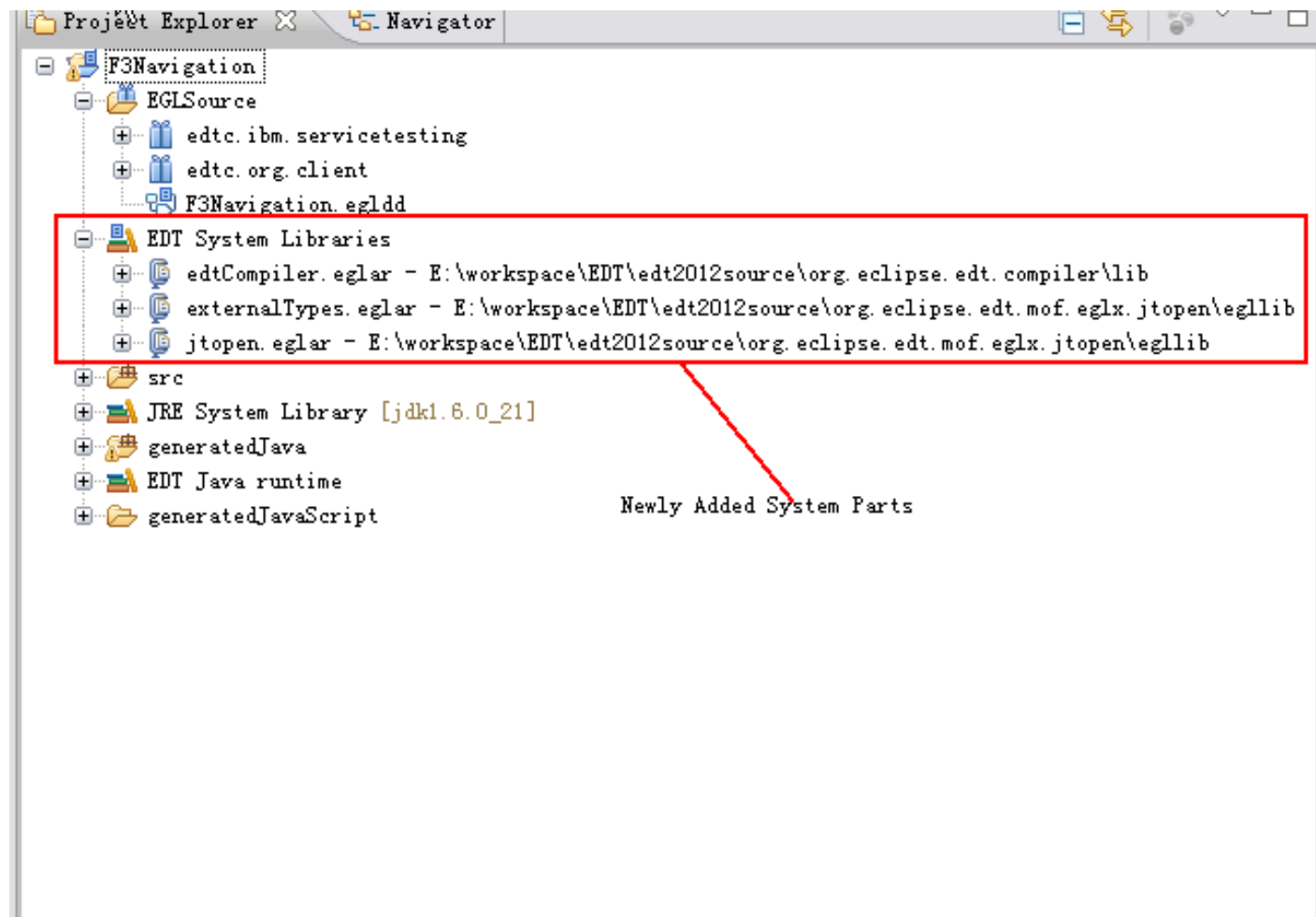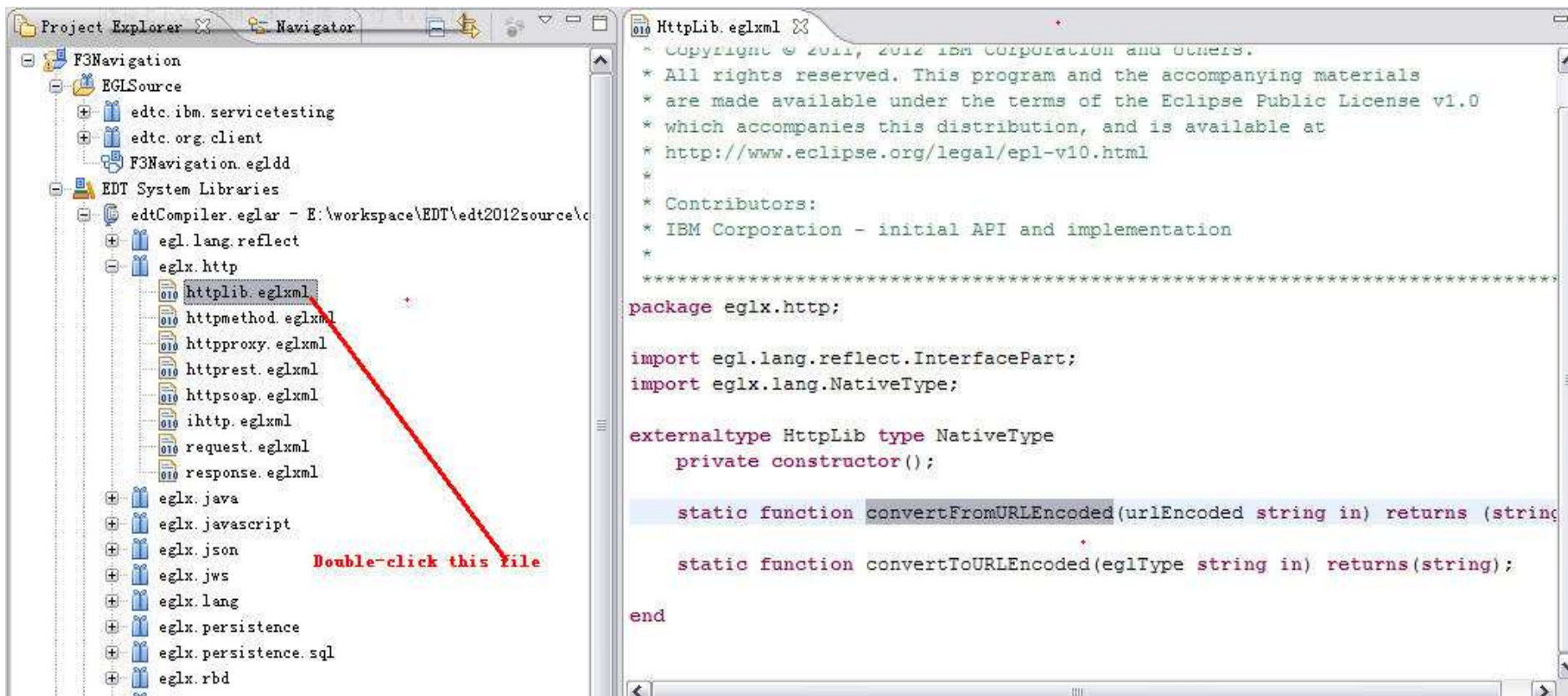
Project Explorer View Update

- You can browse files contained in each EGLAR file

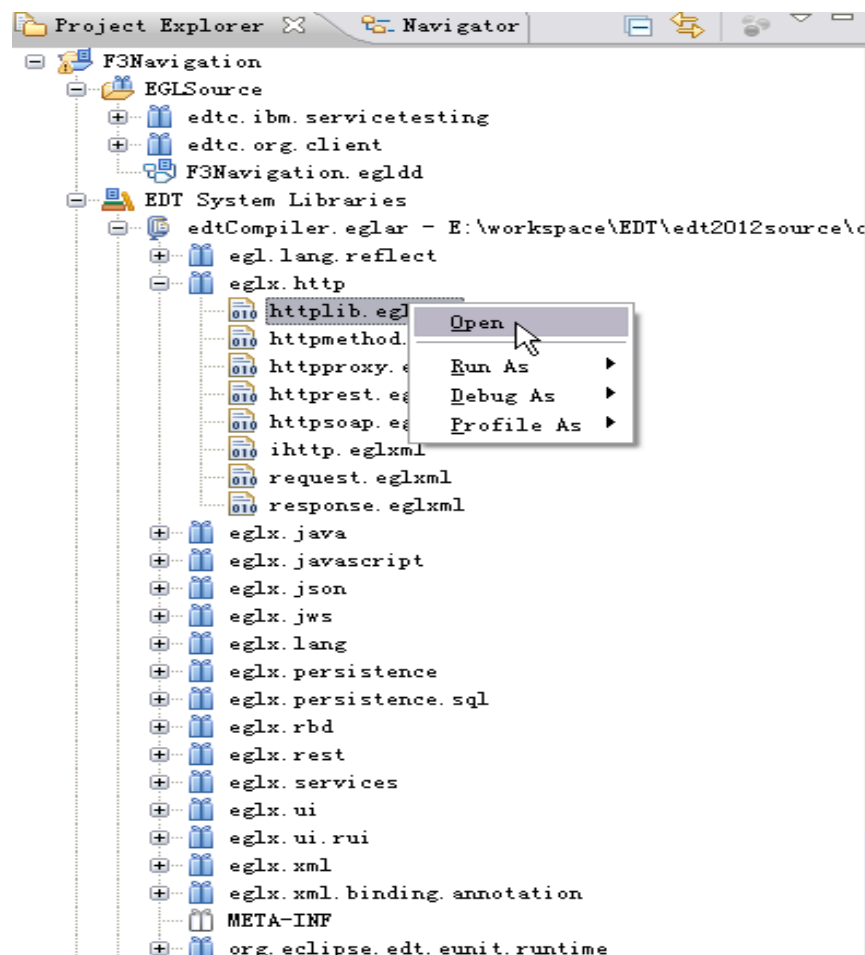# Display IR files in EGL Read-only Editor

- Double-click an IR file, the corresponding EGL source file is opened in read-only editor.
  - Navigate edtCompiler.eglar -->eglx.http --> httplib.eglxml

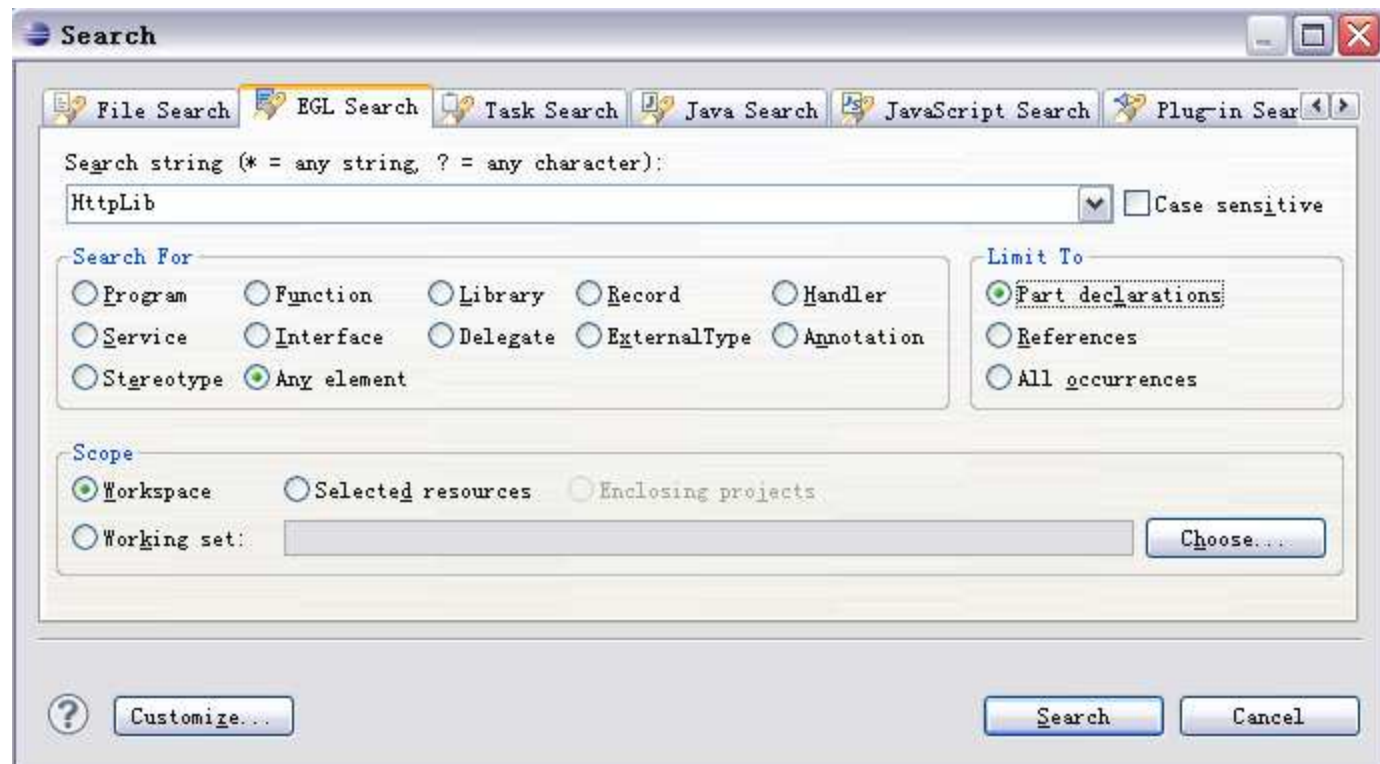# Display IR files in EGL Read-only Editor

- Click an IR file, select Open operation from Context Menu and the corresponding EGL source file is opened in EGL read-only editor.
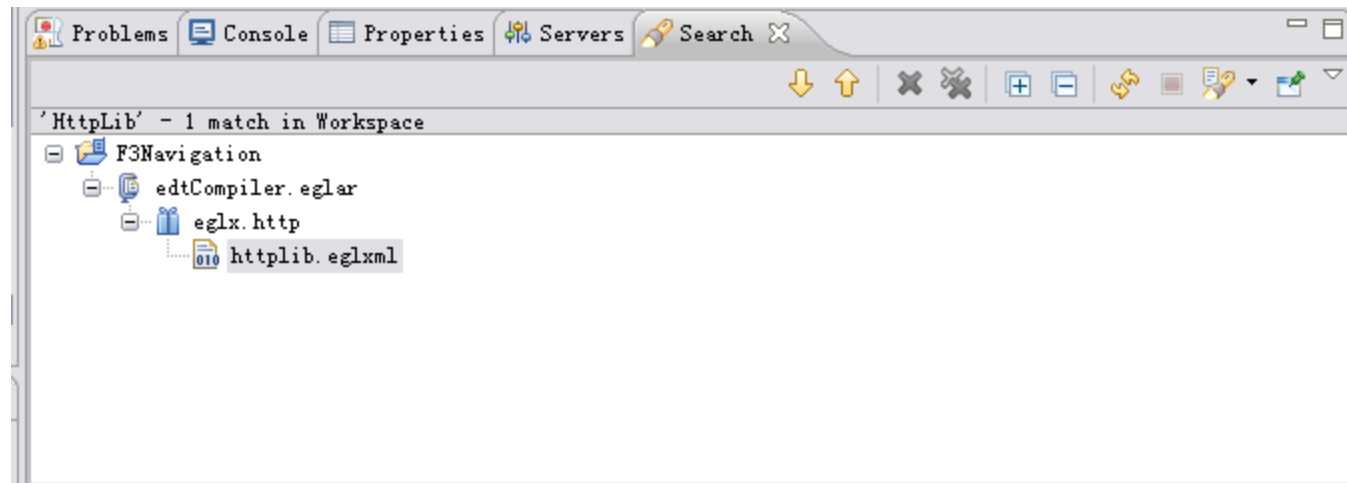
EGLAR Search Support

- EGL Part Search (Part Declaration & Reference)

EGLAR Search Support
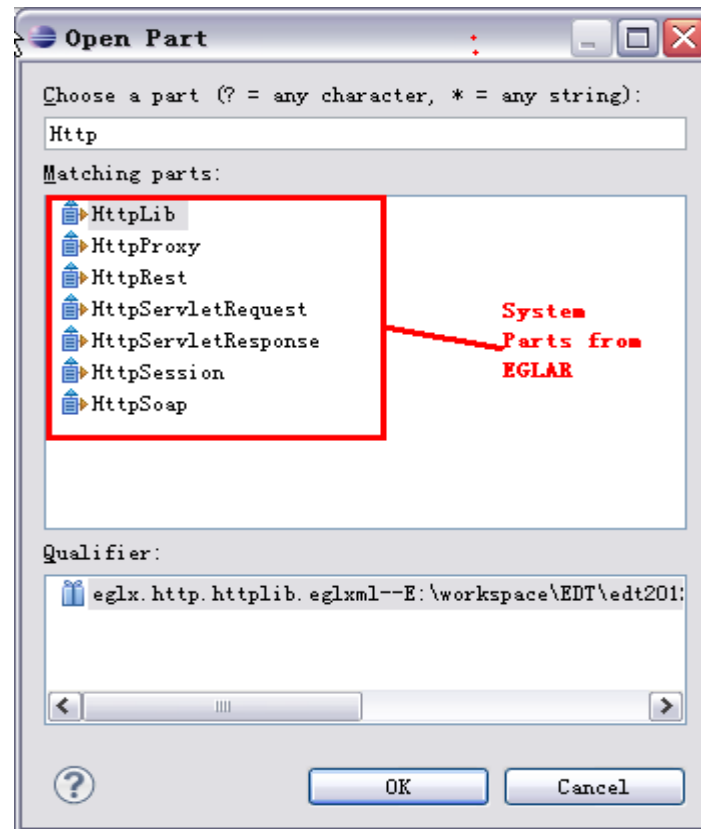
- Matched Part(s) are shown in the Search View

Open System Part through Open Part Wizard

- Open Part by navigating through Navigate --> Open Part, and inputting 'Http'

- The system parts from the EGLAR is shown

- Select 'HttpProxy', and it is opened in read-only editor.

What's New In 0.8.1 M1 / M2

Language Enhancements

Foreach statement update

- Can now be used with an array as the "from" clause.

- Must specify a temporary field to hold the current iteration's value for arrays.

- Note: SQL usage of Foreach is unchanged

- Example:

```
names string[] = ["Bob", "Sue", "Hank"];
  foreach (name string from names)
             syslib.writestdout("Next name: " + name);
end
```

Output:

Next name: Bob

Next name: Sue

Next name: Hank

Integer Literal Change

- Previously the type of an integer literal depended on its length

- Now an integer literal will always be of type "Int"

- An integer literal can be made a "Bigint" by following it with an uppercase "I"

- An integer literal can be made a "Smallint" by following it with an lowercase "i"

- Example:

```
i int = 12345;
b bigint = 12345I;
s smallint = 12345i;
```

- This allows you to control the type, removing unnecessary type conversions.

New primitive types

- Time
  - ▶ Similar to Date and Timestamp, but just the hours, minutes, & seconds
  - ▶ When converting to/from a string, the delimiter between segments is a colon ":" and the hour is in the 24-hour format. Using the wrong format results in a runtime error.
  - ▶ See eglx.lang.ETime in the system parts for a full list of supported functions & operations, including documentation.
  - ▶ Example:
    ```
    t1 time = "15:34:55"; // OK
    t2 time = "15/34/55"; // ERROR - wrong delimiter
    ```

- Limited string - String(n)
  - ▶ Just like String except there is a maximum length allowed on the field
  - ▶ This is NOT a fixed length type
  - ▶ Any text past the maximum length is truncated
  - ▶ Example:
    ```
    s string(5) = "ABCDEFGHIJ"; // holds value "ABCDE"
    ```

What's Coming in M3
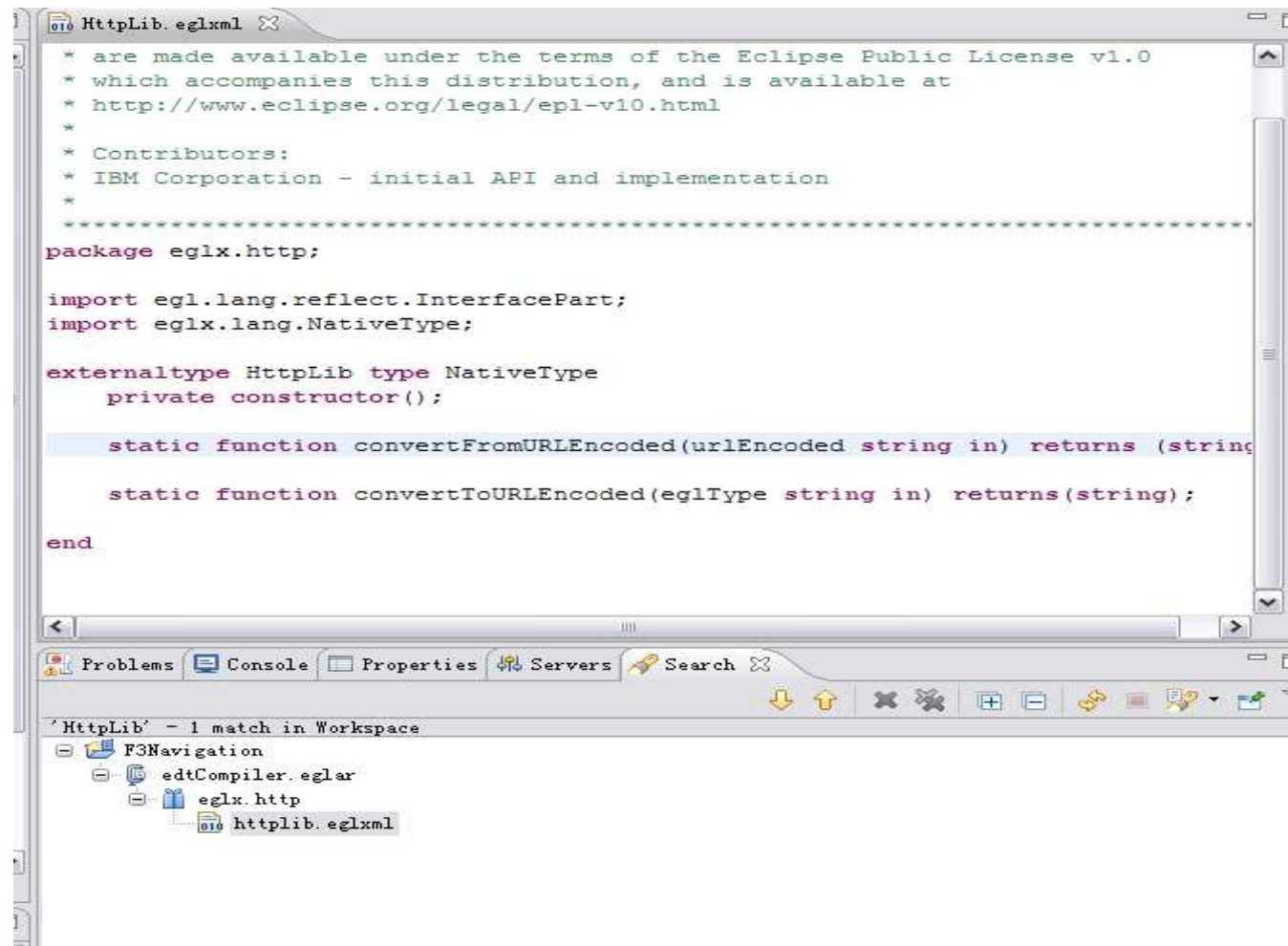
EGLAR Search Support

- Click one of matched part, it will be opened in EGL read-only editor

F3 Navigation Support for System EGLARs

- Navigation from EGL source file to EGL IR file, please refer to below code segment.

  - Click SQLDataSource, and press F3, will navigate to SQLDataSource declaration part.

F3 Navigation Support for System EGLARs

- Navigation from EGL source file to EGL IR file, please refer to below code segment.

   Opened system part.

F3 Navigation Support for System EGLARs

- Navigation from EGL IR file to EGL IR file

    Click DataSource and press F3 in SQLDataSource,IR file for DataSource will be opened.

Vertical Ruler and Line Number Support in Read-only Editor

- Vertical Ruler and Line Number can be shown in read-only editor
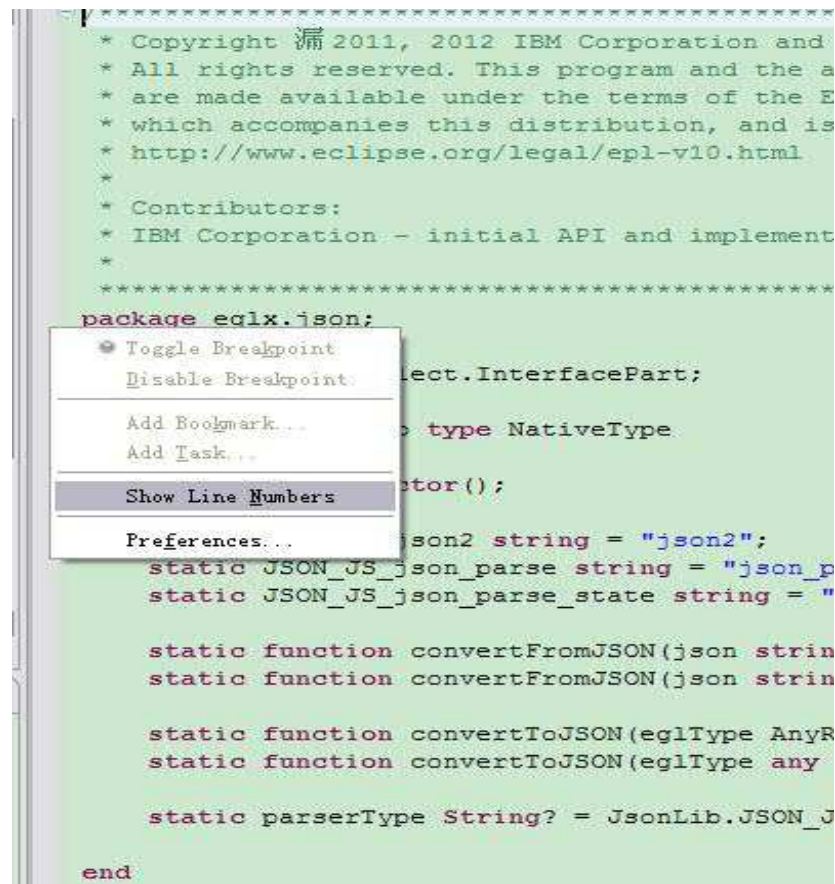
Debugging Support for System EGLARs

- There are no debug limitations for parts in the system EGLARs.

- You can step into the system parts.

- The source code will be displayed and the line is highlighted.

- Breakpoints can be set in the read-only editor.

- The "Run to line" action can be performed in the read-only editor.

- Variables for the system parts will be displayed in the Variables view.

Dojo Mobile Widget Support

- http://wiki.eclipse.org/EDT:Dojo_Mobile_Widgets

- Supported Widgets

  – Button
  – CheckBox
  – List/ListCategory/ListItem
  – Progress
  – Slider
  – Switch
  – Tab Container

  – View/ScrollableView/SwapView
  – IconContainer/IconItem
  – DatePicker
  – TextArea
  – Opener/Overlay

- Mobile Project Template

  – Client Only
  – Client with Services

- Known Restrictions

  – Cannot be used with Desktop Dojo widgets
  – WebKit (Safari) is the recommended for development

Improved Support for Compiler Extensibility

- Support for contributing the following elements to the EDT Compiler:

    - Additional EGL Types

    - Additional runtime files

    - Validation rules

    - Generator Extensions

    - Debug Support

- Supporting for user defined bundles containing the elements listed above

    ▸ Specify supported version ranges for the EDT compiler and generators

    ▸ Single archive files containing Eclipse plug-ins that can be easily consumed by an existing EDT installation

Language Enhancements

- Support for the 'bytes' primitive type

- AS400* data-conversion annotation renaming (see Bug 376979)

- Literals

    - Float

        - ```
          d decimal = 123.45;
          ```

        - ```
          f float = 123.45F;
          ```

        - ```
          s smallfloat = 123.45f;
          ```

    - Byte

        - ```
          b bytes = 0x1A2B3C; // must be lower case x
          ```

What's Coming Beyond M3

EGL Analysis Framework

EGL Analysis Framework (Bug 377473)

- The EGL Analysis Framework provides a way for users to define and execute code analysis rules on EGL source code

- Being developed by Bart van Campenhout at Asist

- The framework will provide:
  - ▸ An engine for running analysis rules
  - ▸ A common UI for displaying, and running rules in the IDE
  - ▸ Support for running rules and displaying results in an SDK environment
  - ▸ Extensible framework for defining additional rules to be written and contributed by the community

- Example Analysis Rules
  - ▸ An upward calltree builder (full tree structure of the programs, services, libraries that are calling (recursively) the current program,library, services)
  - ▸ A downward calltree builder
  - ▸ An extensible EGL programming standards analyzer that will list failing programs, based on naming conventions, sizes of records, sizes of functions, "unallowed" System library functions, …

Getting involved in the EDT Community

EDT Community - Be part of the EDT open-source process!

- Look at EDT's **Bugzilla**, used to track enhancements and bugs
  - ▸ If you're interested in a Bugzilla item, add yourself to the cc: list
  - ▸ Add comments (after signing on with Eclipse ID)

- Ask (and answer) questions on the EDT **Forum**:
  http://www.eclipse.org/forums/index.php?t=thread&frm_id=190

- Familiarize yourself with the EDT **Wiki** documents:
  http://wiki.eclipse.org/EDT

- Read the EDT team **blog**, available via the Community page:
  - ▸ http://eclipse.org/edt/#community
  - ▸ You can also subscribe via Email or Atom feed.

- Follow @Eclipse_EDT on **Twitter**:
  https://twitter.com/#!/Eclipse_EDT

- Charts for this (and previous stakeholder) meetings at:
  - ▸ http://wiki.eclipse.org/EDT:Stakeholder_Meetings

Questions?

Thank You!