Crystal REVS

for Better Results in Less Time

Do you fully understand the code you are working on ?

Answer: If you understood the code **fully**, then there would be no bugs in it.

By increasing your **understanding** of the code,
you can increase the **quality** and **correctness** of your code.

To have *a better understanding* of the code in less time - *Use the right tool for the right task*:

For example:

1. To review a function and the functions called by it, view its CallFlow.
2. To find why a data object contains an incorrect value, examine its Data-dependency tree.
3. To understand a function quickly, view its Flowchart.
4. To examine how a data object is used in the project, view its DataFlow.
5. To view the calling relationship of functions within a file, view the File's function tree.

and so on...

Crystal REVS

for Better Results in Less Time

By using the right tool for the right task,

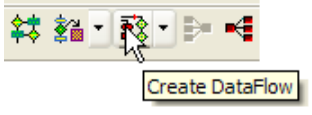
you can have a better understanding of the code in less time.

It will increase the **quality** and **correctness** of your code.

Suppose you are making a design change that affects a particular structure member and you need to examine how that structure member is used in the project.

Use the right tool: View the DataFlow of the structure member.

- **The DataFlow** shows:
  - all the places in the project where that structure member is used
  - and the control flow surrounding those places
- Its graphical view gives you a clear understanding of how the data is used in much less time



Crystal REVS

for Better Results in Less Time

Design changes have better quality and correctness when you use Crystal REVS.

- Before the design change, you can analyze the existing code and affected data in less time.
- After making the design change, check for correctness with DataFlow, Flowcharts etc.

Team discussions are quick and more productive with Flowcharts, Rich Trees, DataFlow, etc.

When you **inherit** legacy code, Crystal REVS will save you a lot of time.

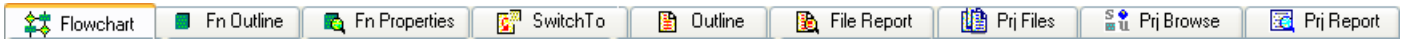
Code Reviews are quick and productive with HTML documentation containing Flowcharts etc.

The graphical views of Flowcharts, DataFlow, CallFlow enable you to think at a higher level.

Everyday, you spend many hours in browsing, understanding and editing code.

With highly effective tools such as: Flowcharts, Rich Trees, DataFlow, CallFlow
Data-dependency trees, Static Analysis, etc.

You can save more than 1 hour every day!



Crystal REVS

Making Better Design Changes in Less Time

- **Before** making changes,

  Crystal FLOW will help you analyze the existing code and affected data objects in less time.

  A few examples:

  1. **How a global or any variable is used** in the project (Use DataFlow)
  2. **What functions are callers of** a given function and **what functions are called by** a given function. (Premium Browsing)
  3. **What parameters are being passed** in the call-sequence up to the current function (Use Rich Tree)

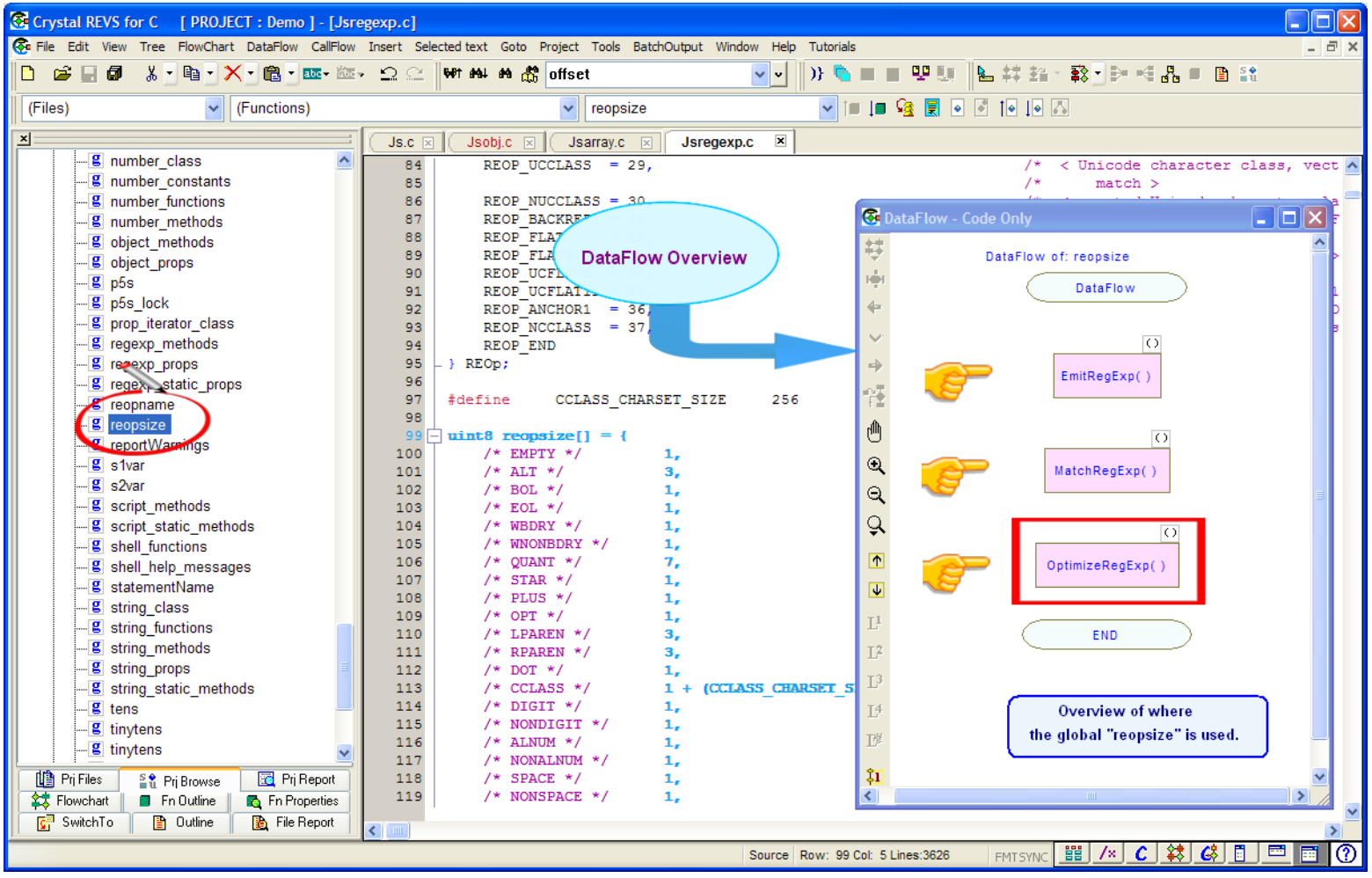
- **After** making code modifications**:**

  - **Check the modified code for correctness** (Use Flowchart, DataFlow)

    (every problem you detect will save much time later)

Use Crystal REVS' auto-formatting of code in real-time. Improve readability. Free yourself from low-level editing.

1. Use DataFlow to examine **how a global variable is used in the project**

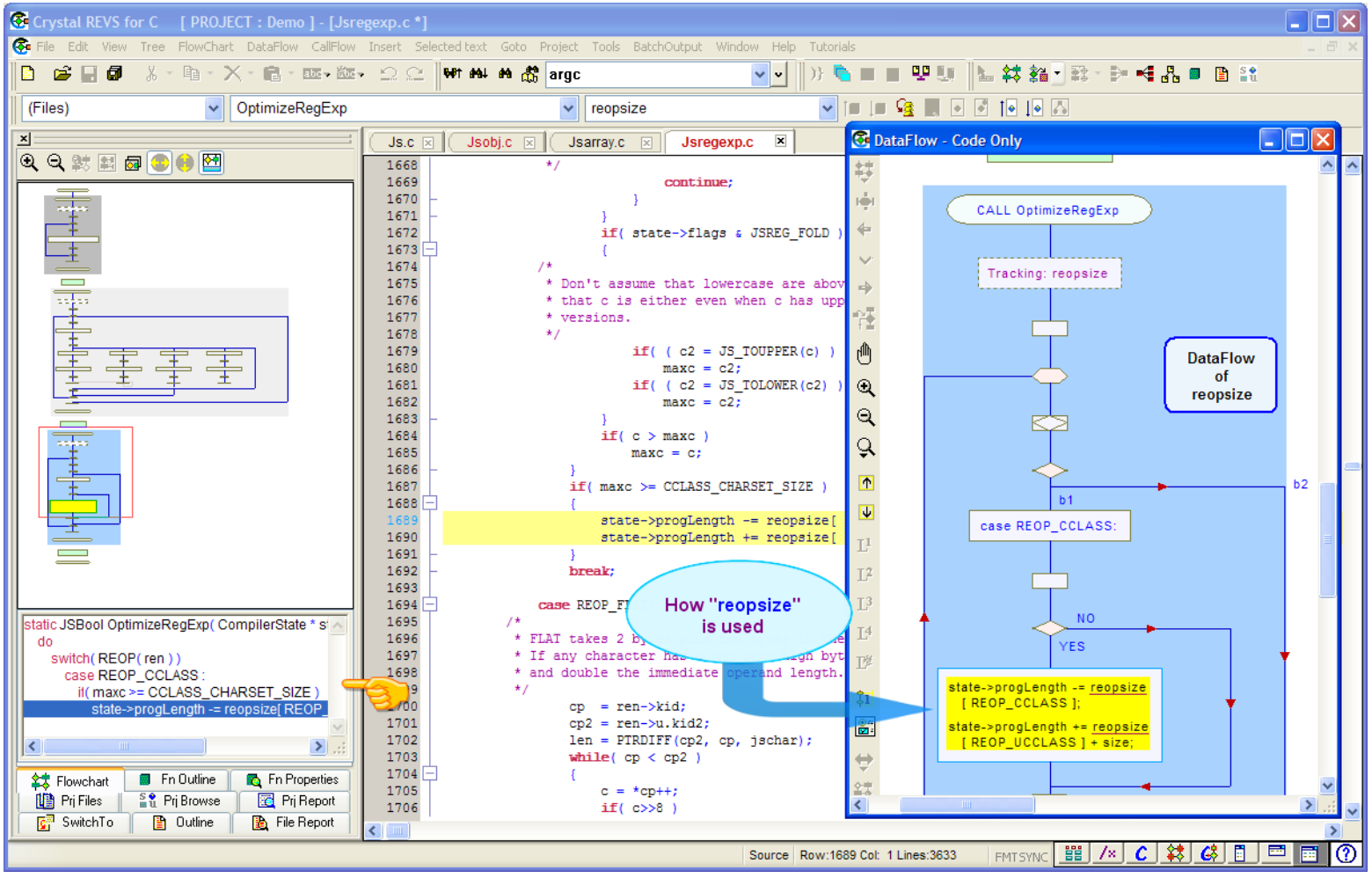


Here the DataFlow Overview shows: the global variable reopsize **is used in three functions - namely, EmitRegExp(), MatchRegExp(), and OptimizeRegExp()**

You are making a design change:

Double-click to expand “OptimizeRegExp()”:

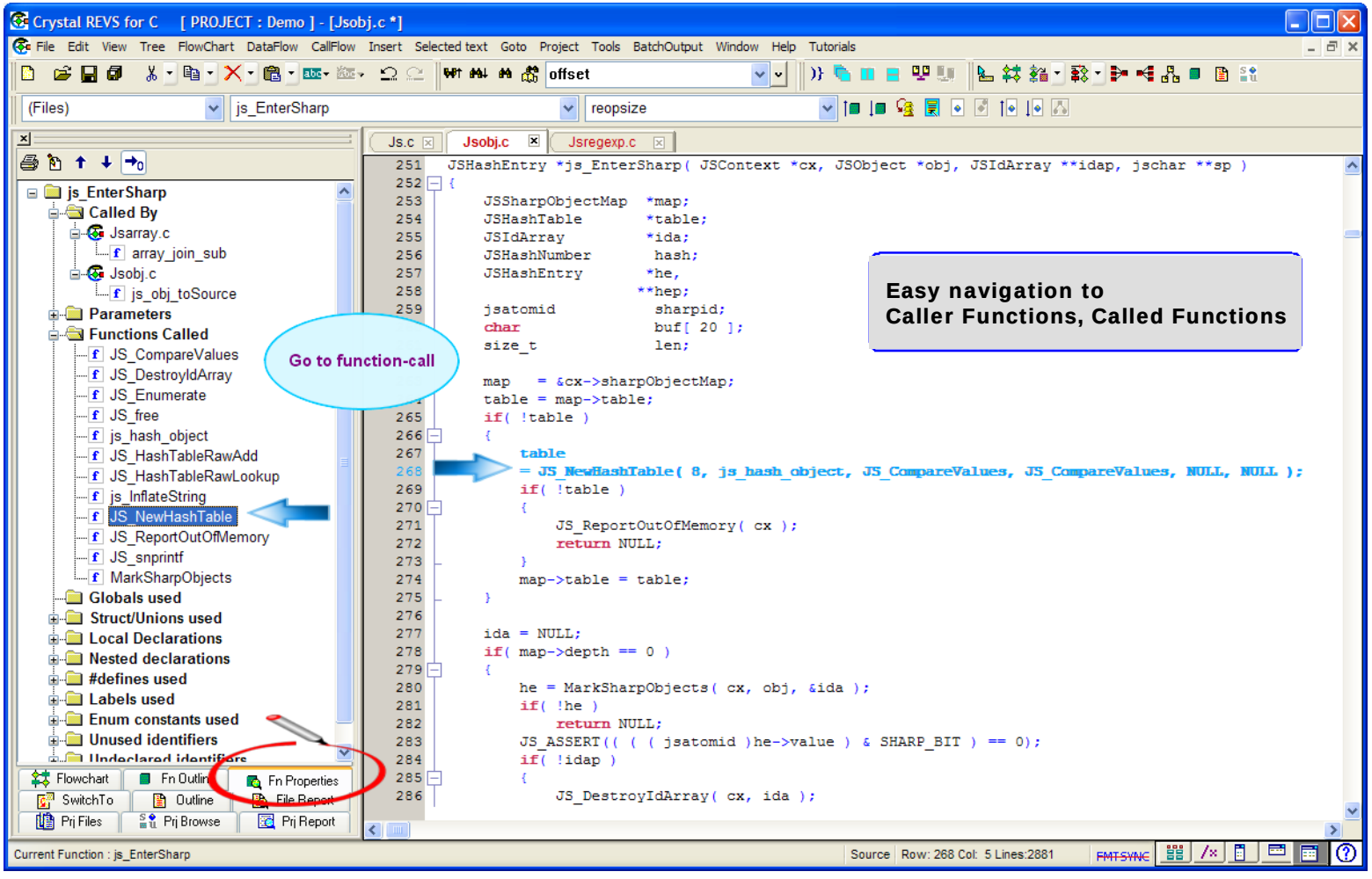
1. The DataFlow expansion shows: **how the global variable** reopsize **is used in** OptimizeRegExp().



You can see the control flow that affects the use of “reopsize”.

You get a better understanding of how data is used - and you save time.

2. View the Function Properties card to view the **Caller functions** and **Called functions**

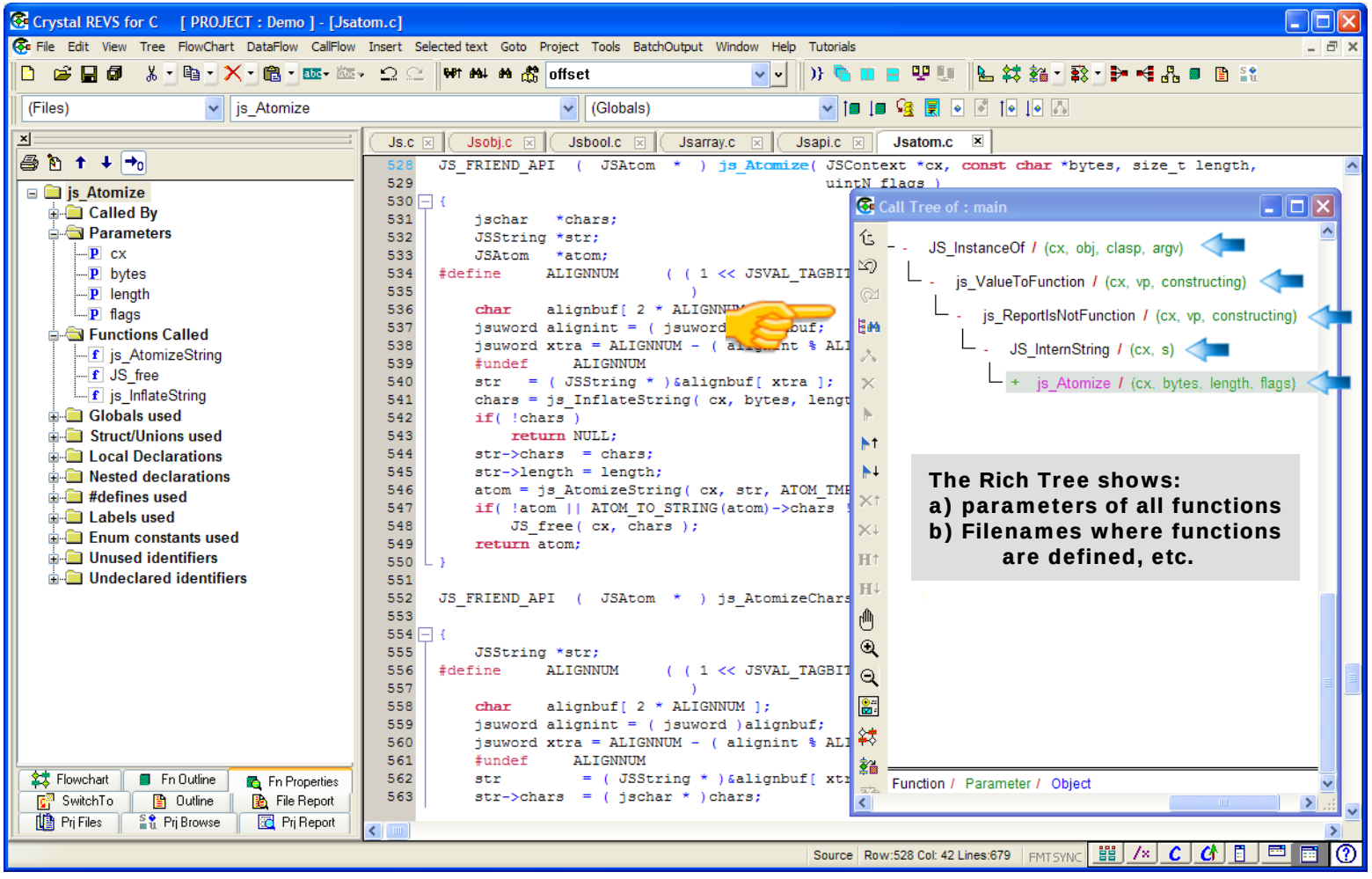


- With single-click, you can go to the function-calls in the code
- With a double-click, you can go to the function's definition

You are making a design change:

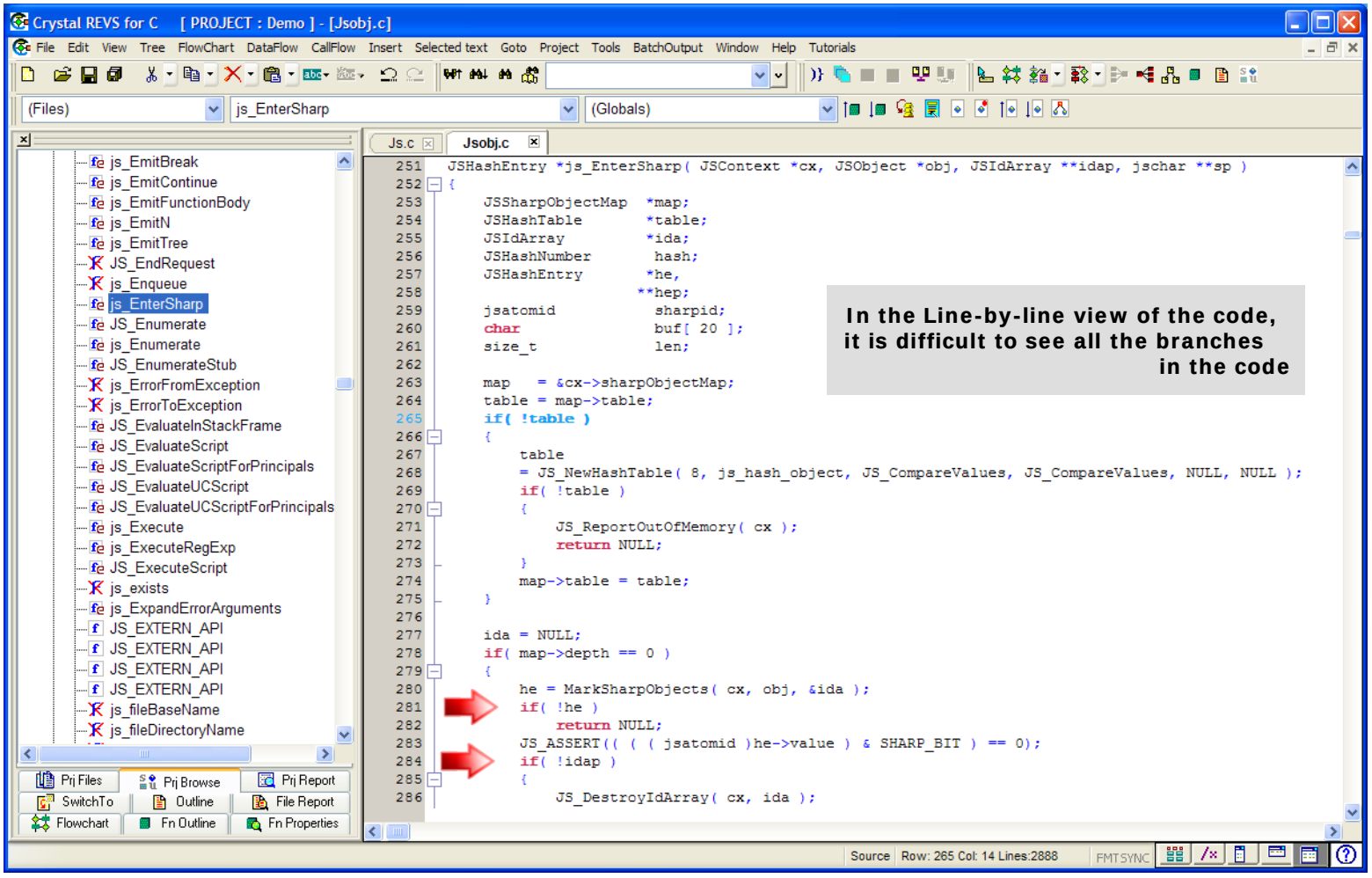
Use the Rich Tree:

View the parameters in the call-sequence



- You can view the parameters of all the functions in the call-sequence.
- You can view the filenames where the functions are defined

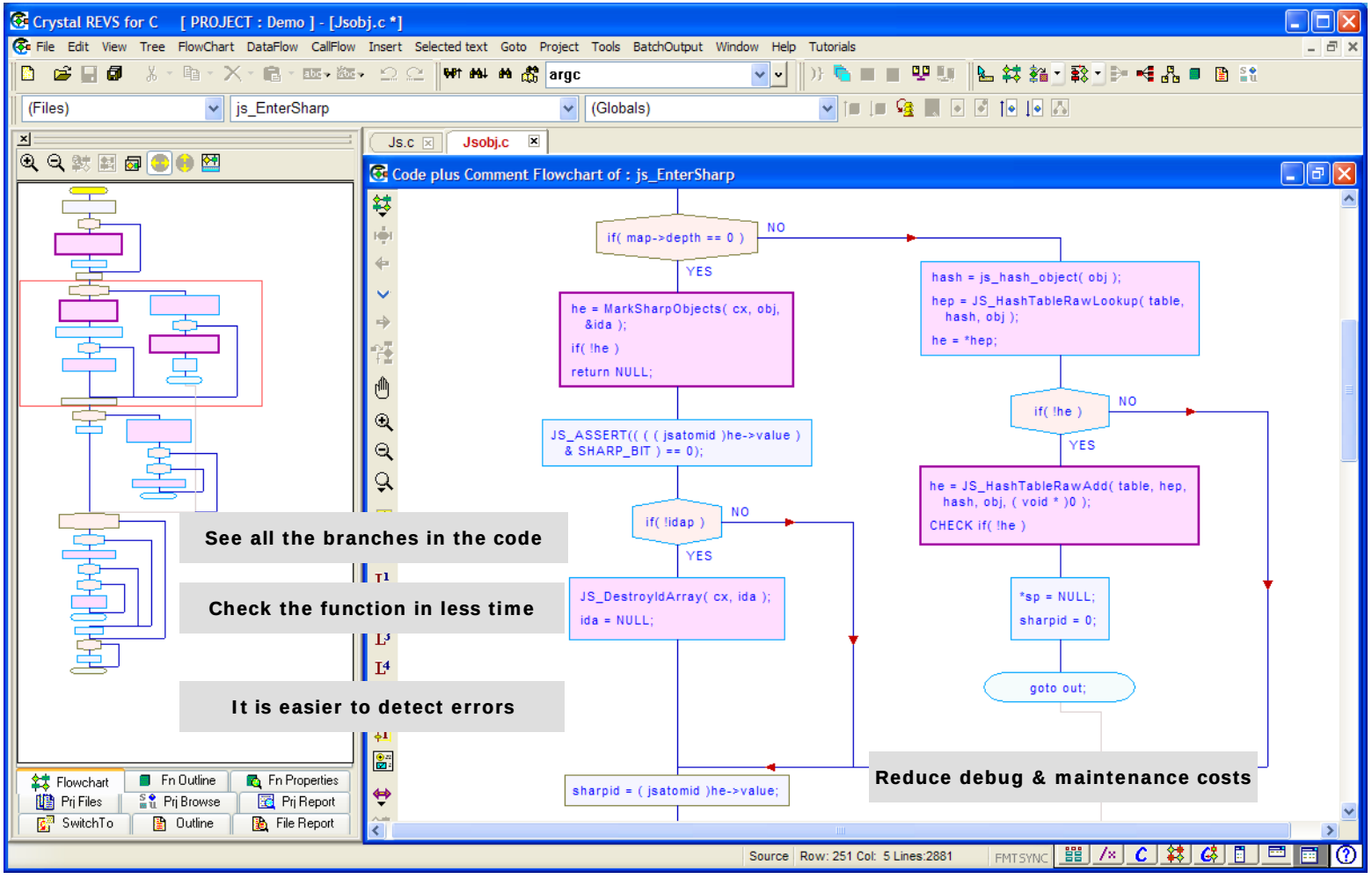
Check the modified code for correctness



The line-by-line view of code is not suitable for checking for correctness.

Here, it is difficult to see all the branches in the code. You can not see the whole function.

With Flowchart, it is easier to check the function for correctness



- You have the whole view of the function
- It is easier to detect errors
- **The flowchart reduces debug and maintenance costs**

In a Team Discussion,

Use Crystal REVS to make the discussion quick and productive

Suppose you are in a team discussion to solve a problem.

1) You are trying to figure out **why a data object has an incorrect value**

Use the **best tool** for the problem:

View the Data-dependency Tree of that data object

2) You are trying to examine **all paths that can reach a given point in a function**

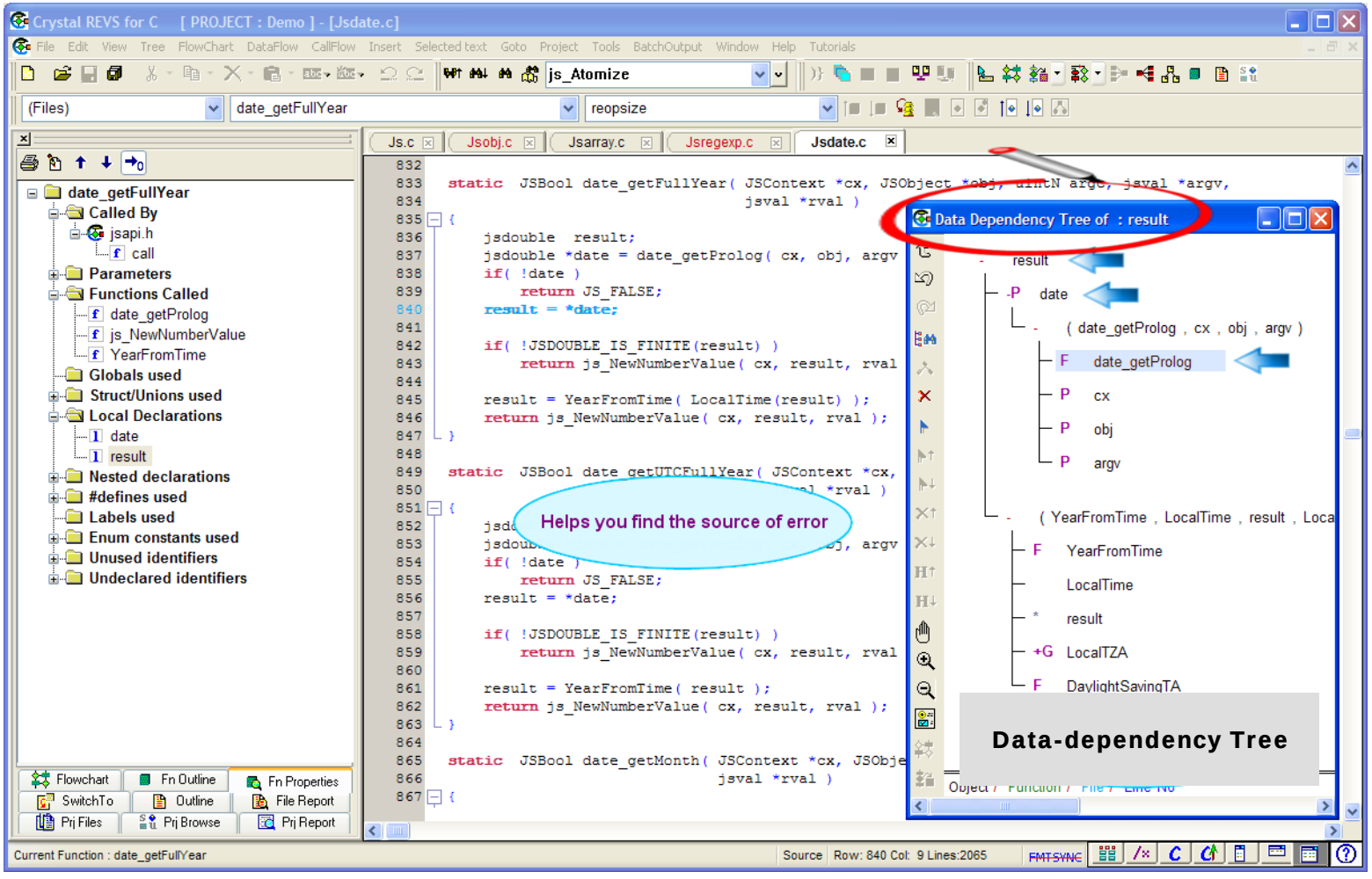
Use the **best tool** for the problem:

View the flowchart and highlight all paths to that point.

Question:

Which would you prefer for team discussion: **Graphical view** *or* **Line-by-line code?**

Data-dependency Tree: To find the source of incorrect data-value



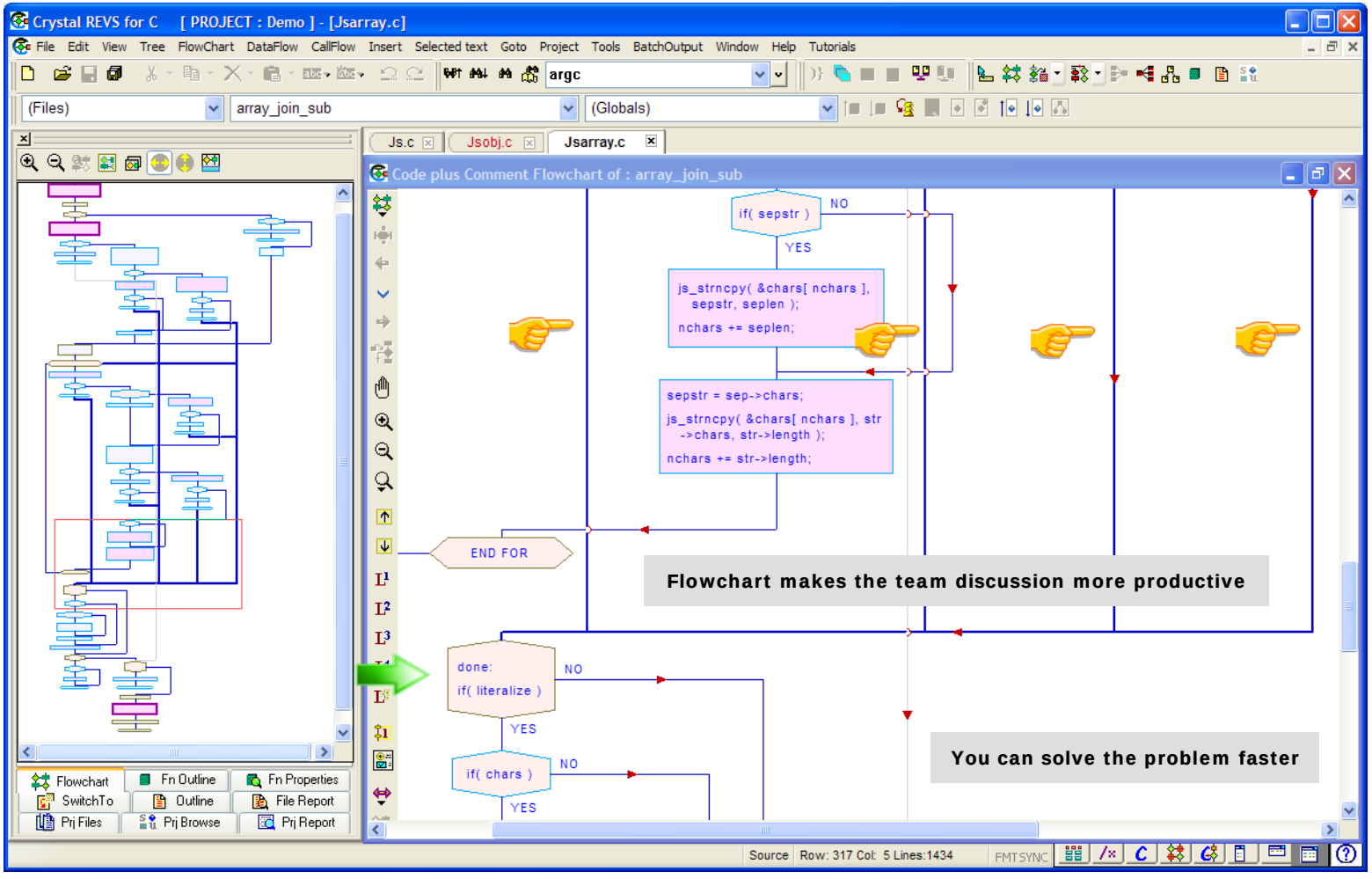
Here we see that the value of "result" **depends** on the value of "date"
which, in turn depends on the function "data_getProlog()"

The Data-dependency tree helps you go to the source of error.

Team discussion to solve a problem:

During the discussion

How do you view **all the paths that reach a given point in the function**



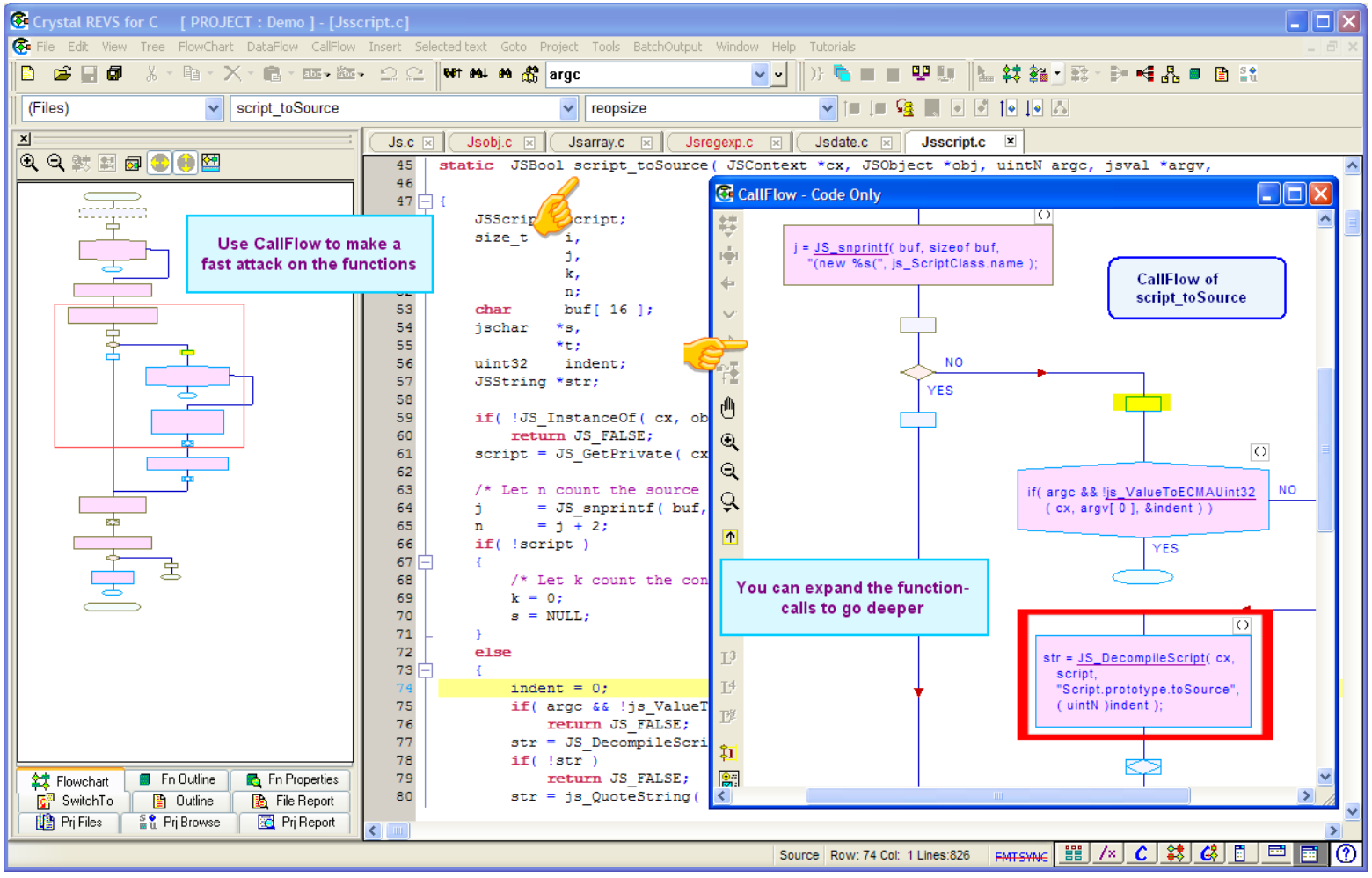
- Use the flowchart to see all the paths that can reach the point of interest
- Flowcharts make the team discussion more effective
- You can find the best solution in less time

Explore and Analyze Code in Less Time

- Do you need to **get an overview of a function** and **the functions called by it ?**
  - **Use CallFlow** - You can expand function-calls and go deeper in the CallFlow

- **Improve readability:**
  - Crystal performs code formatting. **Makes the code easy to read**

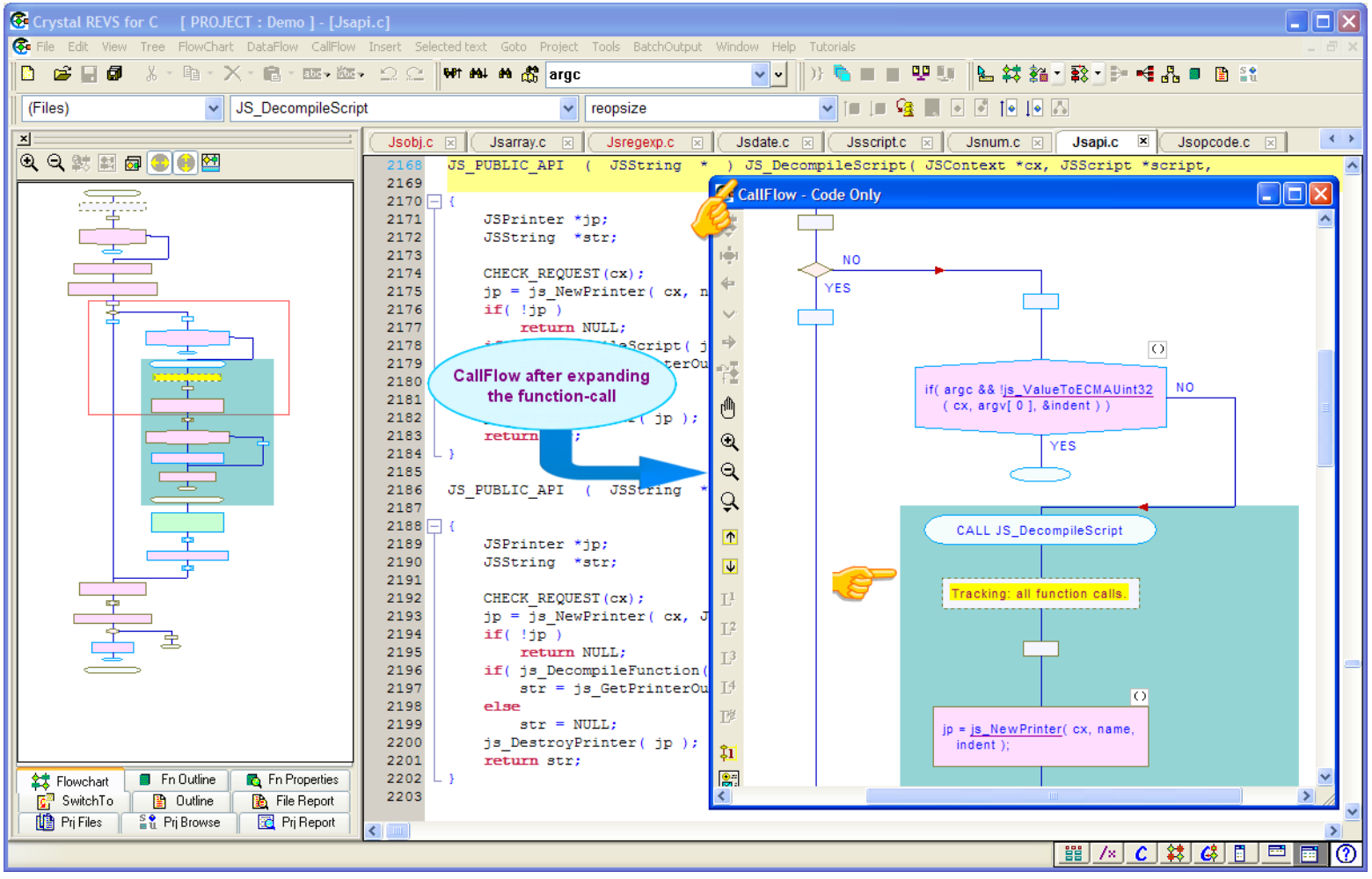
Use CallFlow to make a fast attack on a function and the functions called by it



Here, you see the CallFlow of the function script_toSource( ).

It shows the function-calls and the control flow around the function-calls.

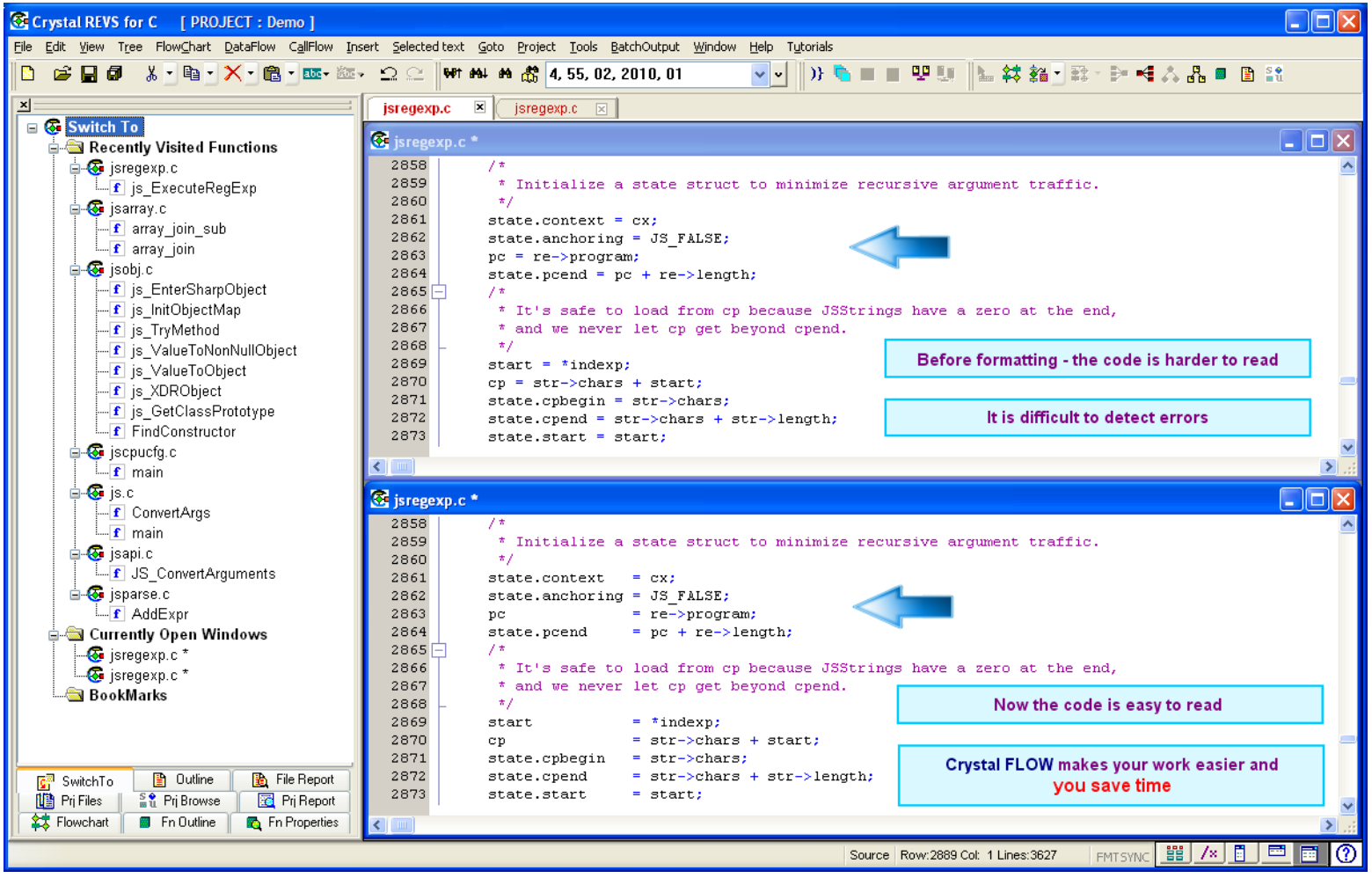
Expand CallFlow to go deeper along the function-calls



- Navigate easily among a set of related function
- You can quickly analyze a function and the functions called by it

You are making a design change:

Crystal performs code formatting and **makes code and comments easy to read**



When you use Crystal, **you save time at every step of your task.**

Use the best tools.

Do a better quality design in less time.

Isn't it the formula to get ahead and stay ahead?

In the global marketplace,

If you lose 1 hour everyday, can you keep up with the competition?

***Get Crystal REVS*. Be ahead of the competition.**

With Crystal REVS, you can save more than **1** hour every day!

Use Crystal REVS. Increase individual productivity. Increase team productivity...

SGV Sarc 907 Broad Oaks Drive, Herndon, Virginia 20170
Phone: 703-904-0678 Fax: 703-904-0155 www.sgvsarc.com