



## A Little Bit About Me

I am the chief technical officer at InVision App, a prototyping and collaboration platform for designers, built by designers. I also rock out in JavaScript and ColdFusion 24x7. ( [more](#) )

Follow me: [Twitter](#) - [Facebook](#) - [LinkedIn](#)



The <http://t.co/JcViV510Ju> job board made its 148th Kiva donation on behalf of "Back-End Web Dev-IT" <http://t.co/etneGJ2XcY> #Jobs #FullTime

Latest via Twitter

[Home](#) [About Me](#) [Projects](#) [Ask Ben](#) [Contact](#) [Job Board](#)



[RSS Feed](#) | [Custom RSS Feed](#)

## The Elvis Operator vs. IsNull() In ColdFusion 11 Beta

Posted February 25, 2014 at 8:24 AM by Ben Nadel

Tags: [ColdFusion](#)

In the beginning, ColdFusion gave us `isDefined()`. Then, `structKeyExists()`. And, as the concept of "null" started to become a first-class concept, ColdFusion delivered unto us `isNull()`. Now, with ColdFusion 11 Beta, we have the "elvis" operator - `?:` - so called because it looks like a well-coiffed emoticon. In the past, we have seen [differences between isDefined\(\) and isNull\(\)](#). As such, I thought it would be good to compare the new Elvis operator to the current `isNull()` function.

In concept, the Elvis operator is something of a short-hand for value parameterization. Meaning, it allows us to "param" a value if it is currently null. The syntax for the operator is:

```
variable = possible_value ?: default_value;
```

If the value in question exists, it is returned as the outcome of the expression (and assigned to the variable on the left). On the other hand, if the value in question does not exist, the default value is returned as the outcome of the expression (and assigned to the variable on the left).

In philosophy, the following examples are all roughly equivalent:

```
1 <!-- NOTE: ColdFusion 11 was in BETA at the time of this writing. -->
2 <cfscript>
3
4     value = ( url.foo ?: "bar" );
5
6     // ----- //
7     // ----- //
8
9     value = ( isNull( url.foo ) ? "bar" : url.foo );
10
11    // ----- //
12    // ----- //
13
14    if ( isNull( url.foo ) ) {
15
```



ColdFusion Engineer

Who We Are: TicketMob is the premier

```
16     value = "bar";
17
18 } else {
19
20     value = url.foo;
21
22 }
23
24 // ----- //
25 // ----- //
26
27 // NOTE: This one is a little different since it's actually
28 // changing the state of url.foo.
29 cfparam( name = "url.foo", default = "bar" );
30
31 value = url.foo;
32
33 </cfscript>
```

philosophy.cfm hosted with [by GitHub](#) [view raw](#)

The concept is simple, but what about the implementation? I've tried to run the Elvis operator through its paces, comparing it to the existing `isNull()` operator. In the first set of examples, I've provided values that should be null.

```
1 <!-- NOTE: ColdFusion 11 was in BETA at the time of this writing. -->
2 <cfscript>
3
4     // Testing unscoped values.
5     elvis = ( theKing ? : "default" );
6     nully = ( isNull( theKing ) ? "default" : "false-positive" );
7
8     writeOutput( "Elvis: " & elvis & "<br />" );
9     writeOutput( "Nully: " & nully & "<br />" );
10
11 </cfscript>
```

fail-1.cfm hosted with [by GitHub](#) [view raw](#)

Gives us the expected:


Elvis: default  
Nully: default

```
1 <!-- NOTE: ColdFusion 11 was in BETA at the time of this writing. -->
2 <cfscript>
3
4     // Testing scoped values.
5     elvis = ( url.theKing ? : "default" );
6     nully = ( isNull( url.theKing ) ? "default" : "false-positive" );
7
8     writeOutput( "Elvis: " & elvis & "<br />" );
9     writeOutput( "Nully: " & nully & "<br />" );
10
11 </cfscript>
```

LOOKING FOR A GREAT JOB?

ticketing and venue management platform. Our forward-thinking entertainment technology company builds progressive tools to help venues, artists and promoters better manage their operations, sell more tickets and know their customers....

Recent Blog Comments




Mark James

Feb 27, 2014 at 3:28 PM

**Encountered "(. Incorrect Select Statement, Expecting a 'FROM', But Encountered '(' Instead, A Select Statement Should Have a 'FROM' Construct.**

I found this same issue when trying to use MID() in a cfquery of a query (PDF Solr Collection). I've been going nuts trying to find away around this problem all day. I need part of the date string in ... [read »](#)




Dave Gruska

Feb 27, 2014 at 2:45 PM

**Lazy Loading Image With AngularJS**

Fantastic code, Ben. Thanks - this really speeds up lazy loading significantly within an ng-repeat. One thing I noticed, though, was that it was taking a while before any images showed up above the ... [read »](#)




Radoslav Sandov

Feb 27, 2014 at 11:47 AM

**Using Underscore.js Templates To Render HTML Partial**

because I wanted to use assignmet of external templates like : `<code> <script type="text/template" id="templateid" src="template.js"></script> <code>` ... [read »](#)




Boyd

Feb 27, 2014 at 11:00 AM

**Preventing Cross-Site Request Forgery (CSRF / XSRF) With AngularJS And ColdFusion**

Thanks man, I love how I never have to search for too long... ..in this case my search was about 10 seconds on google to find yet another useful post from you. lol... You guys rock! :) There is a jQ ... [read »](#)



sergio

Feb 27, 2014 at 10:49 AM

**Using A Name Suffix In ColdFusion's CFMail Tag**

Hello, I am new to coldfusion, I have a problem

fail-2.cfm hosted with [by GitHub](#)

[view raw](#)

Gives us the expected:

*Elvis: default*

*Nully: default*

```
1 <!-- NOTE: ColdFusion 11 was in BETA at the time of this writing. -->
2 <cfscript>
3
4     // Testing bracket notation.
5     elvis = ( url[ "theKing" ] ? "default" );
6     nully = ( isNull( url[ "theKing" ] ) ? "default" : "false-positive" );
7
8     writeOutput( "Elvis: " & elvis & "<br />" );
9     writeOutput( "Nully: " & nully & "<br />" );
10
11 </cfscript>
```

fail-3.cfm hosted with [by GitHub](#)

[view raw](#)

Gives us the ColdFusion error:

*Elvis: default*

*Element theKing is undefined in a Java object of type class coldfusion.filter.UrlScope.*

This is a win for the new Elvis operator! It is able to consume undefined bracket-notation expressions. IsNull(), on the other hand, throws an error.

```
1 <!-- NOTE: ColdFusion 11 was in BETA at the time of this writing. -->
2 <cfscript>
3
4     // Testing deeply-nested values.
5     elvis = ( url.foo.bar ? "default" );
6     nully = ( isNull( url.foo.bar ) ? "default" : "false-positive" );
7
8     writeOutput( "Elvis: " & elvis & "<br />" );
9     writeOutput( "Nully: " & nully & "<br />" );
10
11 </cfscript>
```

fail-4.cfm hosted with [by GitHub](#)

[view raw](#)

Gives us the expected:

*Elvis: default*

*Nully: default*

```
1 <!-- NOTE: ColdFusion 11 was in BETA at the time of this writing. -->
2 <cfscript>
3
4     // Testing strict null values.
5     javaNull = javaCast( "null", "" );
6
7     elvis = ( javaNull ? "default" );
8     nully = ( isNull( javaNull ) ? "default" : "false-positive" );
9
10     writeOutput( "Elvis: " & elvis & "<br />" );
11     writeOutput( "Nully: " & nully & "<br />" );
12
```

with sending mail within the code I want to put the values ??of a specific mail server, SMTP, USER, PASSWORD, PORT. ETC. Thank you. ... [read](#) »



sergio

Feb 27, 2014 at 10:49 AM

### Using A Name Suffix In ColdFusion's CFMail Tag

Hello, I am new to coldfusion, I have a problem with sending mail within the code I want to put the values ??of a specific mail server, SMTP, USER, PASSWORD, PORT. ETC. Thank you. ... [read](#) »



sergio

Feb 27, 2014 at 10:46 AM

### Using A Name Suffix In ColdFusion's CFMail Tag

Hello, I am new to coldfusion, I have a problem with sending mail within the code I want to put the values ??of a specific mail server, SMTP, USER, PASSWORD, PORT. ETC. Thank you. ... [read](#) »



Steve Withington

Feb 27, 2014 at 10:30 AM

### Using The Timeout Attribute With CFHttp In ColdFusion To Limit Blocking

I don't think I've ever used that attribute before ... glad you posted this. Cheers! ... [read](#) »

## ColdFusion Server Monitor

[www.fusion-reactor.com/](http://www.fusion-reactor.com/)

proactive monitoring on CF 8,9 & 10 download FusionReactor today!



```
13 </cfscript>
```

fail-5.cfm hosted with [by GitHub](#)

[view raw](#)

Gives us the expected:

```
Elvis: default
Nully: default
```

```
1 <!-- NOTE: ColdFusion 11 was in BETA at the time of this writing. -->
2 <cfscript>
3
4 // Testing strict null members.
5 request.strictNull = javaCast( "null", "" );
6
7 elvis = ( request.strictNull ? "default" );
8 nully = ( isNull( request.strictNull ) ? "default" : "false-positive" );
9
10 writeOutput( "Elvis: " & elvis & "<br />" );
11 writeOutput( "Nully: " & nully & "<br />" );
12
13 </cfscript>
```

fail-6.cfm hosted with [by GitHub](#)

[view raw](#)

Gives us the expected:

```
Elvis: default
Nully: default
```

```
1 <!-- NOTE: ColdFusion 11 was in BETA at the time of this writing. -->
2 <cfscript>
3
4 // Testing undefined array index.
5 values = [];
6 values[ 2 ] = 2;
7
8 elvis = ( values[ 1 ] ? "default" );
9 nully = ( isNull( values[ 1 ] ) ? "default" : "false-positive" );
10
11 writeOutput( "Elvis: " & elvis & "<br />" );
12 writeOutput( "Nully: " & nully & "<br />" );
13
14 </cfscript>
```

fail-7.cfm hosted with [by GitHub](#)

[view raw](#)

Gives us the ColdFusion error:

```
Elvis: default
Element 1 is undefined in a Java object of type class coldfusion.runtime.Array.
```

This is another win for the new Elvis operator! It is able to consume undefined array elements. IsNull(), on the other hand, throws an error.

```
1 <!-- NOTE: ColdFusion 11 was in BETA at the time of this writing. -->
2 <cfscript>
3
4 // Testing out-of-bounds array index.
5 values = [];
```

```

6
7     elvis = ( values[ 2 ] ): "default" );
8     nully = ( isNull( values[ 2 ] ) ) "default" : "false-positive" );
9
10    writeOutput( "Elvis: " & elvis & "<br />" );
11    writeOutput( "Nully: " & nully & "<br />" );
12
13 </cfscript>

```

fail-8.cfm hosted with [by GitHub](#) [view raw](#)

Gives us the ColdFusion error:

*Elvis: default*  
*The element at position 2 of dimension 1, of array variable "VALUES,"; cannot be found.*

Another win for the new Elvis operator! It is able to consume out-of-bounds array elements. IsNull(), on the other hand, throws an error.

```

1 <!-- NOTE: ColdFusion 11 was in BETA at the time of this writing. -->
2 <cfscript>
3
4     // Testing out-of-bounds list index.
5     list = "a,b,c";
6
7     elvis = ( list.getAt( 5 ) ): "default" );
8     nully = ( isNull( list.getAt( 5 ) ) ) "default" : "false-positive" );
9
10    writeOutput( "Elvis: " & elvis & "<br />" );
11    writeOutput( "Nully: " & nully & "<br />" );
12
13 </cfscript>

```

fail-9.cfm hosted with [by GitHub](#) [view raw](#)

Gives us the ColdFusion error:

*Elvis: default*  
*Invalid list index 5.*

Another win for the new Elvis operator! It is able to consume out-of-bounds list items. IsNull(), on the other hand, throws an error.

```

1 <!-- NOTE: ColdFusion 11 was in BETA at the time of this writing. -->
2 <cfscript>
3
4     // Testing function results.
5     public any function getNothing() {
6         return;
7     }
8
9     elvis = ( getNothing() ): "default" );
10    nully = ( isNull( getNothing() ) ) "default" : "false-positive" );
11
12    writeOutput( "Elvis: " & elvis & "<br />" );
13    writeOutput( "Nully: " & nully & "<br />" );
14
15 </cfscript>

```

fail-10.cfm hosted with [by GitHub](#) [view raw](#)

Gives us the expected:

```
Elvis: default
Nully: default
```

```
1 <!-- NOTE: ColdFusion 11 was in BETA at the time of this writing. -->
2 <cfscript>
3
4 // Testing undefined return values (native methods).
5 values = [];
6 noop = function() {};
7
8 elvis = ( values.each( noop ) ): "default" );
9 nully = ( isNull( values.each( noop ) ) ) ? "default" : "false-positive" );
10
11 writeOutput( "Elvis: " & elvis & "<br />" );
12 writeOutput( "Nully: " & nully & "<br />" );
13
14 </cfscript>
```

fail-11.cfm hosted with [by GitHub](#) [view raw](#)

Gives us the expected:

```
Elvis: default
Nully: default
```

```
1 <!-- NOTE: ColdFusion 11 was in BETA at the time of this writing. -->
2 <cfscript>
3
4 // Testing function results as containers.
5 public any function getNullContainer() {
6     return;
7 }
8
9 elvis = ( getNullContainer().foo ): "default" );
10 nully = ( isNull( getNullContainer().foo ) ) ? "default" : "false-positive" );
11
12 writeOutput( "Elvis: " & elvis & "<br />" );
13 writeOutput( "Nully: " & nully & "<br />" );
14
15 </cfscript>
```

fail-12.cfm hosted with [by GitHub](#) [view raw](#)

Gives us the ColdFusion error:

```
Elvis: default
The system has attempted to use an undefined value, which usually indicates a
programming error, either in your code or some system code. Null Pointers are another
name for undefined values.
```

This is another win for the Elvis operator. It is able to reference keys on undefined return values. IsNull(), on the other hand, throws an error.

```
1 <!-- NOTE: ColdFusion 11 was in BETA at the time of this writing. -->
2 <cfscript>
3
4 // Testing function results as containers.
5 public any function getNullContainer2() {
```

```
6         return;
7     }
8
9     elvis = ( getNullContainer2()[ "foo" ] ? "default" );
10    nully = ( isNull( getNullContainer2()[ "foo" ] ) ? "default" : "false-positive"
11 );
12
13    writeOutput( "Elvis: " & elvis & "<br />" );
14    writeOutput( "Nully: " & nully & "<br />" );
15
</cfscript>
```

fail-13.cfm hosted with [by GitHub](#) [view raw](#)

Gives us the ColdFusion error:

*Elvis: default*  
*The system has attempted to use an undefined value, which usually indicates a programming error, either in your code or some system code. Null Pointers are another name for undefined values.*

Another win for the Elvis operator. Like the previous example, the Elvis operator is able to use bracket-notation to referenced undefined return values. IsNull() continues to throw an error.

```
1 <!-- NOTE: ColdFusion 11 was in BETA at the time of this writing. -->
2 <cfscript>
3
4     // Testing undefined arguments.
5     public any function testArguments( string theKing ) {
6
7         elvis = ( theKing ? "default" );
8         nully = ( isNull( theKing ) ? "default" : "false-positive" );
9
10        writeOutput( "Elvis: " & elvis & "<br />" );
11        writeOutput( "Nully: " & nully & "<br />" );
12
13    }
14
15    testArguments();
16
17 </cfscript>
```

fail-14.cfm hosted with [by GitHub](#) [view raw](#)

Gives us the expected:

*Elvis: default*  
*Nully: default*

```
1 <!-- NOTE: ColdFusion 11 was in BETA at the time of this writing. -->
2 <cfscript>
3
4     // Testing non-enabled sessions.
5     elvis = ( session ? "default" );
6     nully = ( isNull( session ) ? "default" : session );
7
8     writeOutput( "Elvis: " & isStruct( elvis ) & "<br />" );
9     writeOutput( "Nully: " & isStruct( nully ) & "<br />" );
10
11 </cfscript>
```

Gives us the somewhat expected / unexpected:

```
Elvis: YES
Nully: NO
```

Here, we are trying to reference the Session scope in a page that does not have sessions enabled. If you were to try an CFDump the session scope, you would get:

```
The requested scope session has not been enabled.
```

Philosophically, that should make the scope null, which means that both of the above statements should report a non-struct "default" value. However, the Elvis operator seems to believe the session scope exists.

At first, you may think this is a win for the existing isNull() operator. But, you would be misled. In fact, if we turn session-management on such that the session scope does exist, the outcome above does not change. Meaning, the Elvis operator continues to think that the session scope exists and the isNull() operator continues to think that the session scope does not exist.

In this case, both are a fail.

```
1 <!-- NOTE: ColdFusion 11 was in BETA at the time of this writing. -->
2 <cfscript>
3
4 // Testing custom tag's CALLER scope.
5 parent = {};
6
7 cf_tagfail();
8
9 writeOutput( "Elvis: " & elvis & "<br />" );
10 writeOutput( "Nully: " & nully & "<br />" );
11
12 </cfscript>
```

... with the ColdFusion custom tag:

```
1 <cfscript>
2
3 caller.elvis = ( caller[ "parent.member" ] ? "default" );
4 caller.nully = ( isNull( caller[ "parent.member" ] ) ? "default" : "false-
5 positive" );
6
7 </cfscript>
```

Gives us the ColdFusion error:

```
Elvis: default
Element parent.member is undefined in a Java object of type class
coldfusion.runtime.PageScope.
```

This is another win for the Elvis operator, which would work with undefined values in the Caller scope.

Those are all the "null" use-cases that I could think of last night. And, as you can see, the Elvis operator is actually a winning addition in a few of the use-cases.



Now, let's take a quick look at non-null use cases. Meaning, let's look at how the Elvis operator performs when the value in question already exists.

```
1 <!-- NOTE: ColdFusion 11 was in BETA at the time of this writing. -->
2 <cfscript>
3
4 // Testing "undefined" CGI members.
5 elvis = ( form ? : "false-negative" );
6 nully = ( isNull( form ) ? "false-negative" : form );
7
8 writeOutput( "Elvis: " & isStruct( elvis ) & "<br />" );
9 writeOutput( "Nully: " & isStruct( nully ) & "<br />" );
10
11 // NOTE: Long-standing issue.
12 // http://www.bennadel.com/blog/1773-IsNull-vs-IsDefined-For-ColdFusion-9-Scope-
13 Detection.htm
14
15 </cfscript>
```

pass-1.cfm hosted with by GitHub view raw

Gives us the somewhat expected:

Elvis: YES  
Nully: NO

I say "somewhat expected" because it's a long-known issue that isNull() cannot properly identify scopes. I am not sure if this is a win for the Elvis operator, as I am too lazy to perform a request that eliminates the FORM scope. But, given the session-check above, it's entirely possible that neither of the approaches are working properly.

```
1 <!-- NOTE: ColdFusion 11 was in BETA at the time of this writing. -->
2 <cfscript>
3
4 // Testing "undefined" CGI members.
5 elvis = ( cgi.cf11 ? : "false-negative" );
6 nully = ( isNull( cgi.cf11 ) ? "false-negative" : "" );
7
8 writeOutput( "Elvis: " & elvis & "<br />" );
9 writeOutput( "Nully: " & nully & "<br />" );
10
11 </cfscript>
```

pass-2.cfm hosted with by GitHub view raw

Gives the expected:

Elvis:  
Nully:

Both operators correctly return an empty string for any undefined CGI value.

```
1 <!-- NOTE: ColdFusion 11 was in BETA at the time of this writing. -->
2 <cfscript>
3
4 // Testing new "member" functions.
5 value = "ColdFusion 11";
6
7 elvis = ( value.ucase() ? : "false-negative" );
8 nully = ( isNull( value.ucase() ) ? "false-negative" : "" );
9
```

```
10     writeOutput( "Elvis: " & elvis & "<br />" );
11     writeOutput( "Nully: " & nully & "<br />" );
12
13 </cfscript>
```

pass-3.cfm hosted with [by GitHub](#)

[view raw](#)

Gives us the unexpected:

*Elvis: false-negative*

*Nully:*

This is a fail for the Elvis operator. It didn't understand that the result of the native method was a non-null value.

```
1 <!-- NOTE: ColdFusion 11 was in BETA at the time of this writing. -->
2 <cfscript>
3
4     // Testing Java functions.
5     values = [ 1 ];
6
7     elvis = ( values.size() ? "false-negative" );
8     nully = ( isNull( values.size() ) ? "false-negative" : "" );
9
10    writeOutput( "Elvis: " & elvis & "<br />" );
11    writeOutput( "Nully: " & nully & "<br />" );
12
13 </cfscript>
```

pass-4.cfm hosted with [by GitHub](#)

[view raw](#)

Gives us the unexpected:

*Elvis: false-negative*

*Nully:*

Another fail for the Elvis operator, having trouble with member function returns.

```
1 <!-- NOTE: ColdFusion 11 was in BETA at the time of this writing. -->
2 <cfscript>
3
4     // Testing Java functions.
5     request.getSomething = function() {
6         return( true );
7     };
8
9     elvis = ( request.getSomething() ? "false-negative" );
10    nully = ( isNull( request.getSomething() ) ? "false-negative" : "" );
11
12    writeOutput( "Elvis: " & elvis & "<br />" );
13    writeOutput( "Nully: " & nully & "<br />" );
14
15 </cfscript>
```

pass-5.cfm hosted with [by GitHub](#)

[view raw](#)

Gives us the unexpected:

*Elvis: false-negative*

*Nully:*

Another fail for the Elvis operator. Clearly, the Elvis operator doesn't understand how to evaluate the non-null return value for any method, native or otherwise.

```
1 <!-- NOTE: ColdFusion 11 was in BETA at the time of this writing. -->
2 <cfscript>
3
4 // Testing first-class native functions.
5 elvis = ( ucase ? : "false-negative" );
6 nully = ( isNull( ucase ) ? "false-negative" : "" );
7
8 writeOutput( "Elvis: " & elvis & "<br />" );
9 writeOutput( "Nully: " & nully & "<br />" );
10
11 </cfscript>
```

pass-6.cfm hosted with [by GitHub](#) [view raw](#)

Gives us the unexpected:

Elvis: false-negative  
Nully: false-negative

In this case, neither operators correctly identify the first-class native method.

```
1 <!-- NOTE: ColdFusion 11 was in BETA at the time of this writing. -->
2 <cfscript>
3
4 // Testing user defined functions
5 public any function noop() {
6     return;
7 }
8
9 elvis = ( noop ? : "false-negative" );
10 nully = ( isNull( noop ) ? "false-negative" : noop );
11
12 writeOutput( "Elvis: " & isCustomFunction( elvis ) & "<br />" );
13 writeOutput( "Nully: " & isCustomFunction( nully ) & "<br />" );
14
15 </cfscript>
```

pass-7.cfm hosted with [by GitHub](#) [view raw](#)

Gives us the expected:

Elvis: YES  
Nully: YES

At least they both get this one right; while neither operators could identify the first-class native method, they both can properly identify a first-class user-defined function.

```
1 <!-- NOTE: ColdFusion 11 was in BETA at the time of this writing. -->
2 <cfscript>
3
4 // Testing custom tag's CALLER scope.
5 parent = {
6     member: "exists"
7 };
8
9 cf_tagpass();
10
```

```
11 writeOutput( "Elvis: " & elvis & "<br />" );
12 writeOutput( "Nully: " & nully & "<br />" );
13
14 </cfscript>
```

pass-8.cfm hosted with by GitHub [view raw](#)

... with the ColdFusion custom tag:

```
1 <cfscript>
2
3     caller.elvis = ( caller[ "parent.member" ] ? "false-negative" );
4     caller.nully = ( isNull( caller[ "parent.member" ] ) ? "false-negative" : caller[
5 "parent.member" ] );
6
</cfscript>
```

tagpass.cfm hosted with by GitHub [view raw](#)

Gives us the expected:

*Elvis: exists*  
*Nully: exists*

Both operators correctly work with an existing variable name in Caller scope.

Well, I'm running out of time so I'll wrap this up quick. The Elvis operator has a few little bugs, mostly with false-negatives. But, overall, it definitely operators in a wider range of scenarios when compared to its sibling function / operator, isNull().

**NOTE:** Sorry, I didn't really have time to proof-read this post. Damn you day job!

 Like One person likes this. [Sign Up](#) to see what your friends like.

.....

## You Might Also Be Interested In:

- [Robust CFScript Support For Tags In ColdFusion 11 Beta](#)
- [IsNull\(\) vs. IsDefined\(\) For ColdFusion 9 Scope Detection](#)


Looking For a New Job? [View All Jobs](#) | [Post A Job »](#)

- **Back-End Web Developer-Information Technologist** at Michigan State University
- **ColdFusion Developer** at Nonfat Media
- **Mid-to-Senior Level Web Application Developer** at SiteVision, Inc.
- **Junior-to-Mid Level Web Application Developer** at Peavey Electronics
- **Full time on site Senior Coldfusion Developer needed** at Interactive Sites

25% of job board revenue is donated to Kiva. *Loans that change lives* - [Find out more »](#)

.....

## Reader Comments

 **Michael**  
Feb 25, 2014 at 11:48 AM // [reply »](#)

1 Comments  
9 Points

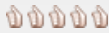
Not 'elvis' but ternery

<http://en.wikipedia.org/wiki/%3F>:



Ben Nadel

Feb 25, 2014 at 12:22 PM // reply »



11,694 Comments

80,148 Points

@Michael,

It looks like when it's used with only two values, it's the Elvis operator (according to the link you sent):

*See also Elvis operator when ?: is used as a null-coalescing binary operator with only two operands.*

So, it looks like it's a specialized form of the ternary operator.

.....

 Like One person likes this. [Sign Up](#) to see what your friends like.

### Post A Comment

**Comment Etiquette:** Please do not post spam. Please keep the comments on-topic. Please **do not post unrelated questions** or **large chunks of code**. And, above all, please be nice to each other - we're trying to have a good conversation here.

Author Name:

Author Email:

Author Website:

Comment:

**Supported HTML tags for formatting:**  
<strong>bold</strong>   <em>Italic</em>   <code>code</code>

Remember my information

Subscribe to comments

Send me a copy of this comment