

# False Positive w/ Pessimistic Value...

---



Guest

When switching value analysis mode to "pessimistic" (Build 1609), I get a false warning with the following code:

```
internal sealed class IndexedEnumerableDebugView collection)
```

```
{  
    if (collection == null)  
        throw new ArgumentNullException("collection");  
    _collection = collection;  
}
```

```
DebuggerBrowsable\(DebuggerBrowsableState.RootHidden\)
```

```
public T[] Items
```

```
{  
    get  
    {  
        var items = new T[_collection.Count];  
        // On the following line, R# incorrectly warns "_collection" may
```

be null.

```
        _collection.CopyTo(items, 0);  
        return items;
```

False Positive w/ Pessimistic Value...

```
    }  
  }  
}
```

Since "\_collection" is a private readonly field in a sealed class, and "Items" is not accessed before the field is assigned, the field cannot possibly be null. Further, a `NullReferenceException` would have already been thrown by the previous statement when accessing "\_collection.Count".



Guest [1](#). **Re: False Positive w/ Pessimistic Value Analysis** Feb 4, 2010 9:30 PM

Addendum: in this case, the "CopyTo" method is an extension method, and the first parameter (the "this" parameter) is annotated `NotNull`:

```
public static void CopyTo(NotNull this IEnumerable source, NotNull  
T[] destination, int destIndex) { ... }
```

"Mike Strobel" <[mike.strobel@gmail.com](mailto:mike.strobel@gmail.com)> wrote in message

news:[hkf3h2\\$cbu\\$1@nntp-server.labs.intellij.net](mailto:hkf3h2$cbu$1@nntp-server.labs.intellij.net)...

When switching value analysis mode to "pessimistic" (Build 1609), I get a false warning with the following code:

>

```
internal sealed class IndexedEnumerableDebugView<T>
{
    private readonly IEnumerable<T> _collection;
```

>

```
public IndexedEnumerableDebugView(IEnumerable<T> collection)
{
    if (collection == null)
        throw new ArgumentNullException("collection");
    _collection = collection;
}
```

>

```
DebuggerBrowsable(DebuggerBrowsableState.RootHidden)
public T[] Items
{
    get
    {
        var items = new T[_collection.Count];
        // On the following line, R# incorrectly warns "_collection"
        may be null.
        _collection.CopyTo(items, 0);
        return items;
    }
}
```

False Positive w/ Pessimistic Value...

```
}  
  
}
```

>

Since "\_collection" is a private readonly field in a sealed class, and "Items" is not accessed before the field is assigned, the field cannot possibly be null. Further, a `NullReferenceException` would have already been thrown by the previous statement when accessing "\_collection.Count".