

# X# Version history

*Note: When an item has a matching GitHub ticket then the ticket number is behind the item in parentheses prefixed with #. You can find these tickets by going to: <https://github.com/X-Sharp/XSharpPublic/issues/nnn> where **nnn** is the ticket number. If you find an issue in X# we recommend that you report it on GitHub. You will be notified of the progress on the work on your issue and the ticket number will be included in the what's new documentation*

*This document lists the changes to X# since build 2.12. For a complete list of changes that were made in earlier builds, please have a look at the help file.*

## Changes in 2.19.0.2

### Compiler

#### Bug fixes

- Now the compiler properly reports an error when duplicate field names are defined in a type (#1385)
- Fixed problem with defining multiple type constraints in a generic type (#1389)
- Fixed problem with global MEMVARs hiding local variables or parameters with the same name (#1294)
- Bogus compiler error messages with not found type (#1396)
- Fixed compiler crash with missing reference to XSharp.VFP in the FoxPro dialect (#1405)
- Fixed problem with /initlocals compiler option incorrectly also initializing class fields (#1408)
- Fixed a problem in the preprocessor where an extended match symbol would not properly match an expression that started with a string literal

#### New features

- We added support for dimensioning (FoxPro) class properties, such as in

```
DIMENSION this.Field(10)
```

- We have added support for FOREACH AWAIT, like in the following example. (Works in .Net Core, .Net 5 and later)

```
FOREACH AWAIT VAR data IN GenerateNumbersAsync(number)  
    SELF:oListView1:Items:Add(data)
```

```
NEXT
```

- We have added support for "Coalescing Member Access, such as in the following example where FirstName and LastName are both properties of the oPerson object:

```
? oPerson:(FirstName+" "+LastName)
```

- The WITH command now also recognizes the AS DataType clause
- XBase++ Class declarations now also allow "END CLASS" as closing token.
- Now the compiler reports an error when attempting to convert from Lambda Expression to usual (#1343)
- We have added support for TUPLE datatypes. This includes declaring local variables, parameters,

return value etc.

We also support decomposition of a tuple return value into multiple locals. See the TUPLE help topic for more information.

## Runtime

### Bug fixes

- Fixed problem calling DoEvents() from the macro compiler (#872 )
- Fixed problem with \_\_Mem2StringRaw() (undocumented) function (#1302)
- Fixed problem opening DBFCDX index file with incorrect collation information in the header #1360
- Fixed problem with OrdSetFocus() resetting the current order when called without arguments (#1362)
- Fixed problems with some index files after a DBPack() (#1367)
- Fixed problem with Deleted() returning TRUE on a table with all records deleted (#1370)
- Fixed problem with opening and writing to a file with FWrite() etc functions when opened in exclusive and write only mode (#1382)
- Fixed several problems (VO incompatibilities) with the SplitPath() function (#1384)
- Now when there is no codepage found in a dbf header, then the DOS codepage from the RuntimeState is used and no longer the hardcoded codepage 437 (#1386)
- Replaced the Dictionary<,> class used in some areas of the runtime with ConcurrentDictionary<,> to avoid issues in multi threaded apps (#1391)
- Fixed problem with NoVarget when using IDynamicProperties (FoxPro dialect) (#1401)
- Fixed problem with Hex2C() giving different results with lower case letters than with upper case. **Note that this bug existed also in VO, so now the behavior of Hex2C() with lower case hex letters in X# with is different to VO (#1402)**
- Accessing properties on a closed DbServer object that was opened with the Advantage RDD could cause problems in the debugger. The DbServer class now returns empty values when the server is closed.

### New features

- Implemented CREATE CURSOR command [FoxPro] (#247). Also implemented CREATE TABLE and ALTER TABLE (FoxPro dialect)
- Implemented INSERT INTO commands (FoxPro dialect for inserting variables from values, arrays, objects and memory variables. INSERT INTO from a SQL query does not work yet.
- Implemented new FoxPro-compatible version of Str() function in XSharp.VFP (#386)
- Now an error is thrown when opening an index file fails (#1358)
- Added AscA() function and made Asc() dependent on the SetAnsi() setting in the runtime (#1376)

## Header files

### Bug fixes

- Implemented several missing commands (#1407)
- Fixed typo in the SET DECIMALS TO command (#1406)
- Added missing clauses NAME and MEMVAR for the GATHER command (FoxPro) (#1409)
- Updated several commands to make some tokens optional and more compatible to various dialects (#1410, #1412)
- Fixed various incompatibilities with COMMIT command in various dialects (#1411)

## Visual Studio Integration

### Bug fixes

- Fixed problem with looking up public static field in type referenced by static using (#1307)
- Fixed intellisense problem with locals defined inside block statements (#1345)
- Fixed problem with Intellisense incorrectly resolving type specified in code with full name to another from the usings list (#1363)
- Fixed problem with member completion incorrectly showing static methods after typing a colon (#1379)
- Fixed editor freezing with specific code (#1380)
- Fixed problem with Class Navigation bar not showing the method name in certain cases (#1381)

### New features

- Added support for IEnumerable and DataTable Debugger visualizers (#1373).
- Please note that when browsing X# arrays the results in the visualizer are really ugly because the visualizer ignores attributes to hide properties and fields for our USUAL class.
- Adjusted the Globals, Workareas etc debugger windows to respect the global theme selected in VS (#1375). Also added status panel to the Workarea window, so you can see the workarea status or field names/values
- Added intellisense support for locals declared with USING VAR or USING (LOCAL) IMPLIED (#1390)
- Now the intellisense database uses an SQLite package that has ARM support, so the it will work also on a Mac and other platforms (#1397)

## VOXporter

### Fixes

- Fixed problem with VOXporter incorrectly modifying previously commented code with {VOXP:UNC} tags (#1404)

## Documentation

### Bug Fixes

- The documentation of functions in the runtime help was describing functions incorrectly. For example the topic title for the "Left" function was "Functions.Left Method" This has been changed to "Left Function"
- The "SingleLineEdit" class in the documentation was called "Real4LineEdit". This has been fixed.

### New Features

- We have added additional documentation to the X# Programming guide about several subjects.

## Changes in 2.18.0.4

## Compiler

### Bug fixes

- Fixed some preprocessor issues with XBase++ related commands (#1213, #1288, #1337)
- Fixed problem with implicit access to static class members (XBase++ dialect) (#1215)
- Fixed a parser error with the DIMENSION command (VFP dialect) (#1267)
- Fixed preprocessor problem with UDCs in code spanning in multiple lines (#1298)

- Now each "unused variable" warning is reported at the exact location of a variable definition, instead of always at the first one (#1310)
- Fixed bogus "unreachable code" warning in the SET RELATION command (#1312)
- Fixed a problem in generating XML documentation for the compiler generated <Module>.\$AppInit and <Module>.\$AppExit methods (#1316)
- Fixed problem with accessing hidden fields of another object (XBase++ dialect) (#1335)
- Fixed problem with calling parent methods with an explicit class indication (Xbase++ dialect) (#1338)
- Fixed problem with incorrectly calling function twice, in code like "SLen(c := SomeFunction())" (#1339)
- Fixed problem with parent Class methods not being visible in derived classes (Xbase++ dialect) (#1349)
- Fixed problem with ::new() not working properly in class methods (Xbase++ dialect) (#1350)
- Fixed a compiler error when returning super:Init() from a XBase++ method (#1357)

## New features

- Introduced warning for not specifying the OUT keyword for OUT parameters (#1295)
- The parser rules for method and constructor calls without parameters have been updated. This may result in a bit faster compilation.
- SLen() is no longer "inlined" by the compiler. If you reference XSharp.Core in your app, SLen() now gets resolved to the SLen() function inside X# Core.
- If you compile without X# runtime, or compile against the Vulcan Runtime you now need to add a SLen() function to your code.
- This is the code inside X# Core that you can use as a template
 

```
FUNCTION SLen(cString AS STRING) AS DWORD
  LOCAL len := 0 AS DWORD
  IF cString != NULL
    len := (DWORD) cString:Length
  ENDIF
  RETURN len
```
- Added support for preprocessor commands #ycommand and #ytranslate that are also supported by Harbour. They work the same as #xcommand and #xtranslate, but the tokens are compared in case sensitive mode (#1314)
- Code generation for some of the XBase++ specific features has changed.
- We have added several more UDCs with the IN <cursor> clause
- We have added UDC support for the FoxPro CAST expression
- Several more SET commands now also support the & operator
- The compiler now supports "Late bound names" in more locations, such as in the REPLACE command, With command etc. This now compiles without problems:
 

```
cVar := "FirstName"
WITH oCustomer
  .&cVar := "John"
END WITH
```

 and this too
 

```
cVar := "FirstName"
REPLACE &cVar with "John"
```

## Runtime

### Bug fixes

- Fixed problem with incorrectly closing dbf file before relations are cleared (#1237)
- Fixed incorrect index scope visibility immediately after file creation (#1238)

- Fixed problem in FFirst()/FNext() not finding all files specified by filter (#1315)
- Fixed problem with DBSetIndex()/VoDbOrdListAdd() always resetting the controlling order to 1 (#1341)
- Fixed problem with updating index keys in the DBFCDX driver when the key expression was of type DATE.
- Fixed a problem when Str() and StrZero() had a built-in maximum string length of 30.(#1352)
- The RegisteredRDD Class now uses a ConcurrentDictionary.
- Fixed a bug in the RDD TransRec() method when a field is missing in the target table (#1372)
- Fixed a problem in the Advantage RDD to prevent ADS functions from being called when the table is closed
- Fixed a problem in the Advantage RDD that could occur when an field with an incorrect name was read
- Fixed a problem in the CurDir() function when the current directory is a UNCPATH (\\Server\Share\SomeDir) (#1378)

## New features

- Added support for accessing indexers in the USUAL type (#1296 )
- We have added a DbCurrency type that is returned from the RDD when a currency field is read.
- Implemented the TEXT TO FILE command (#1304)
- Now the RDD reports an error (dialog) when tagname > maximum length when creating an index order (#1305)
- Added a function \_CreateInstance() that accepts a System.Type parameter
- The late binding code now detects from where Send(), IVarGet() and IVarPut() are called and allow access to private/hidden fields when the calling code is of the same type as the type where the class members were declared. This is used in some of the XBase++ related changes.
- The classes in the XBase++ have been restructured a bit.
- The mapping of several DBF / Workarea / Cursor related UDCs has been changed to be more FoxPro compatible.
- We have added runtime support for the FoxPro CAST expression
- We have done some small code optimizations w.r.t. dictionaries (#1371)
- Several DbServer properties no longer call into the RDD when the server is closed, but return blank values instead.

## Typed SDK classes

- Added a DbServer:Append() overload without parameters (#1320)
- Added missing DataServer:LockcurrentRecord() method (#1321)
- Fixed runtime error when creating a DataWindow with a ShellWindow as owner (#1324)
- Changed DataWindow:Show() method to CLIPPER for compatibility with existing code (#1325)
- Fixed exception when using a ComboBox on a VO Window (#1328)
- Fixed error when opening a datawindow with an assigned server (#1332)
- Fixed runtime error when instantiating a DBServer object with an untyped FileSpec object as first argument (#1348)
- Fixed problem with displaying items in Comboboxes and Listboxes (#1347)
- Several DbServer properties no longer call into the RDD when the server is closed, but return blank values instead.

## Visual Studio Integration

## Bug fixes

- Fixed problem with the "allow dot" setting in the project file (#1192)
- Several macros such as \$CALLSTACK were not returning values in expected format. This has been fixed (#1236)
- Fixed build problem when there is a block comment in the first line of form.prg (#1334)
- Fixed problem with block commenting a code snippet in a single line (#1336)
- Fixed failing project build when the project file contains a property <GenerateAssemblyInfo>True</GenerateAssemblyInfo> (#1344)
- Fixed a problem in the Parser that was causing errors parsing DebuggerDisplay attributes in the expression evaluator.
- The new debugger windows were not following the current windows theme. This is now partially fixed. (#1375)

## VO Compatible Editors

- Fixed design time display issue with CheckBox and RadioButton captions with specific fonts in the VOWED (#796)
- Fixed problem with the VOWED editor changing all existing classes in the prg to PARTIAL (#814)
- Fixed problem with incorrectly adding constructor code to instantiate the DataBrowser in the VOWED, even when there are no (non-deleted) data columns (#1365)
- Fixed several problems in the VOMED with menu item define names in source code and resource files (#1374)

## VOXporter

### New features

- Introduced options (inline in existing code) to comment, uncomment and delete lines from the original VO code (#1303)
  - {VOXP:COM} // comment out line
  - {VOXP:UNC} // uncomment line
  - {VOXP:DEL} and // {VOXP:REM} // remove line

## Installer

### New features

- The installer now detects if the required Visual Studio components "Core Editor" and ".Net Desktop Development" are installed.  
When it finds one or more VS installations but none of these installations has both the required components then a warning is shown.

## Changes in 2.17.0.3

### Compiler

#### Bug fixes

- Fixed several incompatibilities with XBase++ regarding using class members (#1215) **UNCONFIRMED**
- Fixed /vo3 option not working correctly in XBase++ dialect. Also added support for modifiers final, introduce and override (#1244)
- Fixed problem with using the NEW modifier on class fields (#1246)

- Fixed several preprocessor issues with XPP dialect UDCs (#1247, #1250)
- Fixed VO incompatibility with special handling of INSTANCE fields in methods and properties (#1253)
- Fixed problem with the debugger erratically stepping to incorrect lines (#1254, #1264)
- Fixed problem with showing the wrong error line number in some cases with nested statements (#1268)
- Fixed problem where a DO CASE statement without CASE lines was producing an internal error in the compiler (#1281)
- Fixed a couple of preprocessor issues (#1284, #1289)
- Fixed missing compiler error on calling with SUPER a method that does not exist, when late binding is enabled (#1285)
- Fixed a Failed to emit Module error with CONST class field missing value assignment (#1293)
- Fixed a problem with repeated match markers (such as in the SET INDEX TO command) in the preprocessor.
- Fixed a problem that an property definition with an explicit interface prefix could lead to a compiler crash when the interface was "unknown" at compile time and/or the property name was not "Item" (#1306)

## New features

- Added support for "classic" INIT PROCEDURE and EXIT PROCEDURE (#1290)
- Statement blocks without contents now produce a compiler warning (#1281)
- We have made some changes to the lexer and parser in the compiler. This may result in faster compilation speed for code with many nested blocks and a smaller memory footprint.

## Runtime

### Bug fixes

- Fixed several problems (incompatibilities with VO) in CToD() (#1275)
- Added support for 3rd parameter in AAdd() for specifying where to insert the new element (#1287)
- The Default() function now no longer updates usuals that have a value of NULL\_OBJECT to be compatible with Visual Objects. (#1119)
- We have added support for parameters for the AdsSQLServer class (#1282)

## Visual Studio integration

### New Features

- We have added debugger pane windows for the following items:
  - Global variables
  - Dynamic memory variables (Privates and Publics)
  - Workareas
  - Settings
- You can open these windows from the Debug/XSharp menu during debugging. There is also a special "X# Debugger Toolbar" which is also only shown during debugging.
- These windows will only show information when the app being debugged uses the X# runtime (so they will not work in combination with the Vulcan Runtime).
- If you are debugging an application written in another language that uses the X# runtime then these windows will also show information.
- We have planned to add more features to these windows in future builds, like the properties of the current selected area and the field/values in the current selected workarea
- We have added support for "FileCodeModel" for X# files. This is used by the WPF designer and XAML

editor.

This now also fixes the Goto definition in the XAML editor (#1026)

- Several properties of X# projects are now cached. This should result in slightly faster performance.
- We have added support for "Goto Definition" for User Defined commands. For example choosing "Goto definition" on the USE keyword from the USE command will bring you to its definition in our standard header file.

## Bug fixes

- Fixed member completion issue with Type[,] arrays (#980)
- Fixed missing member completion in class inside namespace when same named class exists without namespace (#1204)
- Fixed an auto indent problem when an entity has an attribute in the precessing line (#1210)
- Fixed intellisense problems with static members in some cases (#1212)
- Fixed some intellisense issues with code or declarations spanning in multiple lines (#1221, #1260)
- Fixed intellisense problem with nested classes inside a namespace (#1222)
- Fixed incorrect resolving of VAR local type, when using a type cast (#1224)
- Fixed several problems with collapsing/expanding code in the editor (#1233)
- Fixed showing of bogus member completion list with unknown types (#1255)
- Fixed some problems with auto typing text with Ctrl + Space (complete Word) (#1256)
- Fixed coloring of Text .. EndText statements (#1257)
- Fixed several issues with tooltip hints with generic types (#1258, #1259, #1273)
- Fixed problem with delegate signature not showing in intellisense tooltips (#1265)
- Fixed invalid coloring of code with multiline comments (#1269)
- Fixed invalid entries in member completion after typing "self." (#1270)
- Fixed problem with calling the disassembler when path specified (in option X# Custom Editors\Other Editors\Disassembler) with spaces (#1271)
- Fixed editor coloring completely stopping when using some UDC calls (#1272)
- Fixed problem with hint not showing on CONSTANT locals in FOR statements (#1274)
- Fixed auto indent problem when code contains a LOOP or EXIT keyword (#1278)
- Fixed an exception in the editor when typing a parenthesis under specific circumstances (#1279)
- Fixed problem with incorrectly trying to open in design mode files with filenames starting with an opening bracket (#1292 )
- The "XSharp Website" menu option inside VS was broken (#1297)
- Fixed problem with the Match Identical Identifiers functionality that could slow down Visual Studio
- Fixed a VS lock up that could happen when a file was opened during debugging.
- Parameter tips for classes with a static constructor and a normal constructor were not processed correctly. This has been fixed.
- When a project was opened where the dependency between a dependent item (like a .resx file or a .designer.prg file) and its parent was missing, then an exception could occur, which prevented the project from opening. This has been fixed.
- When 2 compiler errors occurred on the same line with the same error code they were sometimes shown in the VS output window but not in the Error List. This has been fixed (#1308)



## Changes in 2.16.0.5

### Compiler

#### New Features Xbase++ dialect

- We have made several changes in the way how Xbase++ class definitions are generated. Please check your code extensively with this new build!
- We now generate a class function for all classes. This returns the same object as the ClassObject() method for Xbase++ classes.  
This class function is generated, regardless of the /xpp1 compiler option.  
The Class function depends on the function \_\_GetXppClassObject and the XSharp.XPP.StaticClassObject class that both can be found in the XSharp.XPP assembly (#1235).  
From the Class function you can access class variables and class methods.
- In Xbase++ you can have fields (VAR) and properties (ACCESS / ASSIGN METHOD) with the same name, even with same visibility. Previously this was not supported.  
The compiler now automatically makes the field protected (or private for FINAL classes) and marks it with the [IsInstance] attribute.  
Inside the code of the class the compiler will now resolve the name to the field. In code outside of the class the compiler will resolve the name to the property.
- For derived classes the compiler now automatically generates a property with the name of the parent class, that is declared as the parent class and returns the equivalent to SUPER.
- We have fixed an issue with the FINAL, INTRODUCE and OVERRIDE keywords for Xbase++ methods (#1244)
- We have fixed some issues with accessing static class members in the XBase++ dialect (#1215)
- You can now use the "::" prefix to access class variables and class methods inside class methods.
- When a class is declared as subclass from another class then the compiler generates a (typed) property in the subclass to access the parent class, like Xbase++ does. This property returns the value "super".
- We are now supporting the READONLY clause for Vars and Class Vars. This means that the variable must be assigned in the Init() method (instance variables) or InitClass() method (Class vars)

#### New Features other dialects

- Inside Visual Objects you could declare fields with the INSTANCE keyword and add ACCESS/ASSIGN methods with the same name as the INSTANCE field.  
In previous builds of X# this was not supported.  
The compiler now handles this correctly and resolves the name to the field in code inside methods/properties of the class and resolves the name to the property in code outside of the class.
- The PPO file now contains the original white space from user defined commands and translates.

#### Bug fixes

- Fixed some method overload resolution issues in the VO dialect (#1211).
- Fixed internal compiler error (insufficient stack) with huge DO CASE statements and huge IF ELSEIF statements (#1214).
- Fixed a problem with the Interpolated/Extended string syntax (#1218).
- Fixed some issues with incorrectly allowing accessing static class members with the colon operator or instance members with the dot operator (#1219, #1220).
- Fixed Incorrect visibility of MEMVARs created with MemVarPut() (#1223).
- Fixed problem with \_DLL FUNCTION with name in Quotes not working correctly (#1225).
- If the preprocessor generated date and/or datetime literals, then these were not recognized. This has

been fixed (#1232).

- Fixed a problem with the preprocessor matching of the last optional token (#1241).
- Fixed a problem with recognizing the ENDSEQUENCE keyword in the Xbase++ dialect (#1243).
- Using a default parameter value of NIL is now only supported for parameters of type USUAL. Using NIL for other parameter types will generate a (new) warning XS9117.  
Also assigning NIL to a Symbol or using NIL as parameter to a function/method call that expects a SYMBOL will now also generate that warning (#1231)
- Fixed a problem in the preprocessor where two adjacent tokens were not merged into one token in the result stream. (#1247)
- Fixed a problem in the preprocessor where the preprocessor was not detecting an optional element when the element started with a Left parenthesis (#1250)
- Fixed a problem with interpolated strings that contained literal double quotes like in `i"SomeText"{iNum}" " "`
- Fixed a problem that was introduced in 2.16 with local functions / procedures.
- A warning generated at parse time could lead to another warning about a preprocessor define even when that is not needed. This has been fixed.
- Fixed issue with default parameter values for parameters declared as `"a := NIL,b := NIL as USUAL"` introduced in an earlier build of 2.16.
- Fixed issue with erratic debugger behavior introduced in an earlier build of 2.16.
- When you are referring to a type in an external assembly that depends on another external assembly, but you did not have a reference to that other external assembly, then compilation could fail without proper explanation. Now we are producing the normal error that you need to add a reference to that other assembly.
- Omitting the type for a parameter for a function or method that does not have the CLIPPER calling convention is allowed. These parameters are assumed to be of type USUAL. This now produces a new warning XS9118.

## Breaking changes

- If you are using our parser to parse source code, please check your code. We have made some changes to the language definition for the handling of if ... else statements as well as for the case statements (a new condBlock rule that is shared by both rules). This removes some recursion in the language. Also, some of the Xbase++ specific rules have been changed. Please check the language definition online

## Runtime

### New Features

- Added the DOY() function
- Adding missing ADS\_LONG and ADS\_LONGLONG defines
- Improved the speed of CDX skip operations on network drives (#1165)

### Bug fixes

- Fixed a problem with DbSetRelation() and RLock() (#1226).
- Adjusted implicit conversion from NULL\_PSZ to string to now return NULL instead of an empty string.
- Some initialization code is now moved from \_INIT procedures to the static constructor of the SQLConnection Class, in order to make it easier to use this class from non-X# apps.
- Fixed an issue with the visibility of dynamic memory variables that were created with the MemVarPut function (#1223).
- Fixed a problem with the DbServer class in exclusive mode (#1230).
- Implicit conversions from NULL\_PSZ to string were returning an empty string and not NULL (#1234)

- Improved the speed of CDX skip operations on network drives (#1165).
- Fixed a problem in the CTOD() function when the day, month or year were prefixed with spaces
- Fixed an issue with OrderListAdd() in the ADS RDD. When the index is already open, then the RDD no longer returns an error.
- Fixed an issue with MemRealloc where the second call on the same pointer would return NULL\_PTR (#1248).

## VOSDK

- Global arrays in the SDK classes are now initialized from the class constructor of the SqlConnection class to fix problems when the main app does not include a link to the SQL Classes assembly.

## Visual Studio integration

### Debugger

- The debugger expression evaluator now also evaluates late bound properties and fields (if that compiler option is enabled inside your project).
- If this causes negative side effects then you can disable that in the "Tools/Options Debugging/X# Debugger options screen".
- The debugger expression evaluator now is initialized with the compiler options from your main application (if that application is an X# project). The settings on the Debugger Options dialog are now only used when debugging DLLs that are loaded by a non-X# startup project.
- The debugger expression evaluator now always accepts a '.' character for instance fields, properties and methods, regardless of the setting in the project options.
- This is needed because several windows in the VS debugger automatically insert '.' characters when adding expressions to the watch window or when changing values for properties or fields.

### New Features

- Added support for importing Indexes in the DbServer editor.
- The X# project system now remembers which Windows were opened in the Windows editor in design mode and reopens them correctly when a solution is reopened.
- We have added templates for a Harbour console application and Harbour class library.
- We have added item templates for FoxPro syntax classes and Xbase++ syntax classes.
- The Class templates for the FoxPro and XBase++ dialect now include a class definition in that dialect.
- We have improved the support for PPO files in the VS Editor.
- We have updated some of the project templates.

### Bug fixes

- Fixed a problem with incorrectly showing member list in the editor for the "[:=" operator (#1061)
- Fixed VOMED generation of menu item DEFINE names that were different to the ones generated by VO (#1208)
- Fixed VOWED incorrect order of generated lines of code in some cases (#1217)
- Switched back to our own version of Mono.Cecil to avoid issues on computers that have the Xamarin (MAUI) workload in Visual Studio.
- Fixed a problem opening a form in the Form Designer that contains fields that are initialized with an XBase function call (#1251).
- Windows that were in [Design] mode when a solution is closed, are now properly opened in [Design] mode when the solution is reopened.

## Changes in 2.15.0.3

### Compiler

#### New Features

- Implemented the STACKALLOC syntax for allocating a block of memory on the stack (instead of the heap) (#1084)
- Added ASYNC support to XBase++ methods (#1183)

#### Bug fixes

- Fixed missing compiler error in a few specific cases when using the dot for accessing instance members, when /allowdot is disabled (#1109)
- Fixed some issues with passing parameters by reference (#1166)
- Fixed some issues with interpolated strings (#1184)
- Fixed a problem with the macro compiler not detecting an error with incorrectly accessing static/instance members (#1186)
- Fixed incorrect line number reported for error messages on ELSEIF and UNTIL statements (#1187)
- Fixed problem with using an iVar named "Value" inside a property setter, when option /cs is enabled (#1189)
- Fixed incorrect file/line info reported in error message when the Start() function is missing (#1190)
- Fixed bogus warning about ambiguous methods in some cases (#1191)
- Fixed a preprocessor problem with nested square brackets (#1194)
- Fixed incorrect method overload resolution in some cases in the VO dialect (#1195)
- Fixed erratic debugging while stepping over code in some cases (#1200)
- Fixed a problem where a missing "end keyword", such as ENDIF, NEXT, ENDDO was not reported when the code between the start and end contained a compiler warning (#1203)
- Fixed a problem in the build system where sometimes an error message about an incorrect "RuntimeIdentifier" was shown

### Runtime

#### Bug fixes

- Fixed runtime error in DBSort() (#1196)
- Fixed error in the ConvertFromCodePageToCodePage function
- A change in the startup code for the XSharp.RuntimeState could lead to incorrect codepages

### Visual Studio integration

#### New Features

- Added VS option for the WED to manually adjust the x/y positions/sizes in the generated resource with multipliers (#1190)
- Added new options page to control where the editor looks for identifiers on the Complete Word (Ctrl+Space) command.
- A lot of improvements to the debugger expression evaluator (#1050). Please note that this debugger expression evaluator is only available in Visual Studio 2019 and later
- Added a debugger options page that controls how expression are parsed by the new debugger expression evaluator.  
You can also change the setting here that disallows editing while debugging.

- We have added context help to the Visual Studio source code editor. When you press F1 on a symbol then we inspect the symbol. If it comes from X# then the relevant page in the help file is opened. When it comes from Microsoft then we open the relevant page from the Microsoft Documentation online.  
In a next build we will probably add an option for 3rd parties to register their help collections too.
- When a keyword is selected in the editor that is part of a block, such as CASE, OTHERWISE, ELSE, ELSEIF then the editor will now highlight all keywords from that block.
- The Jump Keywords EXIT and LOOP are now also highlighted as part of the repeat block that they belong to.
- When a RETURN keyword is selected in the editor, then the matching "Entity" keyword, such as FUNCTION, METHOD will be highlighted too.
- Added a warning to the Application project options page, when switching the target framework.

## Bug fixes

- Fixed previously broken automatic case synchronization, when using the cursor keys to move to a different line in the editor (#722)
- Fixed some issues with using Control+Space for code completion (#1044, #1140)
- Fixed an intellisense problem with typing ":" in some cases (#1061)
- Fixed parameter tooltips in a multiline expressions (method/function calls) (#1135)
- Fixed problem with Format Document and the PUBLIC modifier (#1137)
- Fixed a problem with Go to definition not working correctly with multiple partial classes defined in the same file (#1141)
- Fixed some issues with auto-indenting (#1142, #1143)
- Fixed a problem with not showing values for identifiers in the beginning of a new line when debugging (#1157)
- Fixed Intellisense problem with LOGICs in some cases (#1185)
- Fixed an issue where the completionlist could contain methods that were not visible from the spot here the completionlist was shown (#1188)
- Fixed an issue with the display of nested types in the editor (#1198)
- Cleaned up several X# project templates, fixing problems with incorrect placement of Debug/Output folders (#1201)
- Undoing a case synchronization in the VS editor was not working, because the editor would immediately synchronize the case again (#1205)
- Rebuilding the intellisense database no longer restarts Visual Studio (#1206)
- VOXporter now writes the menu ids from VO menus to the exported .xsmnu files and these are reused inside X# (#1207)
- A Change to our project system and language service could lead to broken "Find in Files" functionality in some versions of Visual Studio. This has been fixed.
- Fixed an issue where goto definition was not working for protected or private members
- Fixed an issue where for certain files the Dropdown combo boxes on top of the editor were not correctly synchronized.

## Documentation

### Changes

- Some methods in the typed SDK were documented as Function. They are now properly documented as Method
- Property Lists and Method lists for classes now include references to methods that are inherited from parent classes. Methods that are inherited from .Net classes, such as ToString() from System.Object

are NOT included.

## Changes in 2.14.0.2, 3 & 4

### *Visual Studio Integration*

#### **Bug fixes**

- Fixed an exception in the X# Editor when opening a PRG file in VS 2017
- Selecting a member from a completion list with the Enter key on a line immediately after an entry that has an XML comment could lead to extra triple slash (///) characters to be inserted in the editor
- The triple slash command to insert XML comments was not working. This has been fixed.
- Fixed a problem with entity separators not shown on the right line for entities with leading XML comments
- Fixed a peek definition problem with types in source code that do not have a constructor
- Fixed a problem with the Implement Interface action when the keyword case was not upper case
- Fixed a problem that the keyword case was prematurely synchronized in the current line.
- Fixed a problem with indenting after keywords such as IF, DO WHILE etc
- Fixed a problem with selecting words at the end of a line when debugging
- Fixed a problem where Format Document could lock up VS
- Fixed a problem that accessors such as GET and SET were not indented inside the property block
- Fixed a problem that Format Document was not working for some documents
- Changed the priority of the background scanner that is responsible for keyword colorization and derived tasks inside VS.

## Changes in 2.14.0.1

### *Compiler*

#### **Bug fixes**

- Fixed a problem with date literals resulting in a message about an unknown alias "gloal" (#1178)
- Fixed a problem that leading 0 characters in AssemblyFileVersion and AssemblyInformationalVersion were lost. If the attribute does not have the wildcard '\*' then these leading zeros are preserved (#1179)

### *Runtime*

#### **Bug fixes**

- The runtime DLLs for 2.14.0.0 were marked with the TargetFramework Attribute. This caused problems. The attribute is no longer set on the runtime DLLs (#1177)

## Changes in 2.14.0.0

### *Compiler*

#### **Bug fixes**

- Fixed a problem resolving methods when a type and a local have the same name (#922)
- Improved XML doc messages for methods implicitly generated by the compiler (INITs, implicit constructors) (#1128)
- Fixed an internal compiler error with DELEGATEs with default parameter values (#1129)
- Fixed a problem with incorrect calculation of the memory address offset when obtaining a pointer to

a structure element (#1132)

- Fixed problematic behavior of #pragma warning directive unintentionally enabling/.disabling other warnings (#1133)
- Fixed a problem with marking the complete current executing line of code while debugging (#1136)
- Fixed incompatible to VO behavior with value initialization when declaring global MEMVAR (#1144)
- Fixed problem with compiler rule for DO not recognizing the "&" operator (#1147)
- Fixed inconsistent behavior of the ^ operator regarding narrowing conversion warnings (#1160)
- Fixed several issues with CLOSE and INDEX UDC commands (#1162, #1163)
- Fixed incorrect error line reported for error XS0161: not all code paths return a value (#1164)
- Fixed bogus filename reported in error message when the Start() function is missing (#1167)
- The PDB information for a command defined in a UDC now highlights the entire row and not just the first keyword
- Fixed a problem in the CLOSE ALL and CLOSE DATABASES UDC.

## Runtime

### New Features

- Added 2 new values to the DbNotificationType enum: BeforeRecordDeleted and BeforeRecordRecalled. Also added AfterRecordDeleted and AfterRecordRecalled which are aliases for the already existent RecordDeleted and RecordRecalled (#1174)

### Bug fixes

- Added/updated several defines in the Win32API SDK library (#696)
- Fixed a problem with "SkipUnique" not working correctly (#1117)
- Fixed an RDD scope problem when the bottom scope is larger than the highest available key value (#1121)
- Fixed signature of LookupAccountSid() function in the Win32API SDK library (#1125)
- Improved exception error message when attempting to use functions like Trim() (which alter the key string length) in index expressions (#1148)
- Fixed a Macro Compiler runtime exception when there is an assignment in an IIF statement (#1149)
- Fixed a problem with resolving the correct overloaded method in late bound calls (#1158)
- Fixed a problem with parametrized SQLExec() statements in the FoxPro Dialect
- Fixed a problem in the Days() function where the incorrect number of seconds in a day was used.
- Fixed a problem in the Advantage RDD when a FieldGet returned fields with trailing 0 characters. These are now replaced with a space.
- Fixed a problem with DBI\_LUPDATE in the ADS RDDs
- Fixed the Debugger display of the USUAL type.

## Visual Studio integration

### New Features

- Now using the "Reference Manager" instead of the "Add Reference Dialog Box" for adding References (#21, #1005)
- Added an option to the Solution Explorer context menu to split a Windows Form in a form.prg and form.designer.prg (#33)
- We have added an options page to the Tools / Options TextEditor/X# settings that allows you to enable/disable certain features in the X# source code editor, such as "Highlight Word", "Brace Matching" etc.
- Tooltips for all source code items now contain the Location (file name and the line/column).



## Bug fixes

- Fixed a problem renaming files when a solution is under SCC with Team Foundation Server (#49)
- The WinForms designer now ignores differences in the namespaces specified in the form.prg and designer.prg files (the one from form.prg is used) (#464)
- Fixed incorrect mouse tooltip for a class in some cases (#871)
- Fixed a code completion issue on enum types with extension methods (#1027)
- Fixed some intellisense problems with enums (#1064)
- Fixed a problem with Nuget packages in VS 2022 causing first attempts to build projects to fail (#1114)
- Fixed a formatting problem in XML documentation tooltips (#1127)
- Fixed a problem with including bogus extra static members in the code completion list in the editor (#1130)
- Fixed problem with Extension methods not included in Goto Definition, Peek definition, QuickInfo tips and Parameter Tips (#1131)
- Fixed a problem in determining the correct parameter number for parameter tips when a compiler pseudo function such as IIF() was used inside the parameter list (#1134)
- Fixed a problem with selecting words with mouse double-click in the editor with underscores while debugging (#1138)
- Fixed a problem with evaluating values of identifiers with underscores in their names while debugging (#1139)
- Fixed identifier highlighting causing the VS Editor to hang in certain situations (#1145)
- Fixed indenting of generated event handler methods in the WinForms designer (#1152)
- Fixed a problem with the WinForms designer duplicating fields when adding new controls (#1154)
- Fixed a problem with the WinForms designer removing #region directives (#1155)
- Fixed a problem with the WinForms designer removing PROPERTY declarations (#1156)
- Fixed a problem that the type lookup for locals was failing in some cases (#1168)
- Fixed a problem where the existence of extension methods in code was causing a problem filling the member list (#1170)
- Fixed a problem when completing the member completion list without selecting an item (#1171)
- Fixed a problem with showing member completion on types of static members of a class (#1172)
- Fixed a problem with the indentation after single line entities, such as GLOBAL, DEFINE, EXPORT etc. (#1173)
- Optional tokens in UDCs were not colored as Keyword in the source code editor
- Fixed a problem in the CodeDom provider that failed to load on a Build Server because of a dependency to Microsoft.VisualStudio.Shell.Design version 15.0 when generating code for WPF projects.

The What's new for older builds can be found in the [X# documentation](#)