

Frequently Asked Questions on Eiffel ENViSioN!™

We highly recommend that you first take a look at our FAQ's document on the [Eiffel Language](#).

[What is Eiffel ENViSioN! ?](#)

[What is the difference between Eiffel, Eiffel ENViSioN! and EiffelStudio™?](#)

[What does Eiffel ENViSioN! bring to Visual Studio .NET?](#)

[How is Eiffel ENViSioN! different from other .NET languages?](#)

[What operating systems does Eiffel ENViSioN! run on?](#)

[Why the emphasis on reusability with Eiffel ENViSioN! ?](#)

[Is Eiffel ENViSioN! compatible with other .NET languages?](#)

[Is Eiffel ENViSioN! compatible with EiffelStudio?](#)

[Who should use Eiffel ENViSioN! ?](#)

[Won't I have to forsake my existing software, thus losing millions of dollars?](#)

[Are there performance issues with Eiffel for .NET?](#)

[Is Eiffel transparent in Visual Studio .NET?](#)

[What is BON?](#)

What is Eiffel ENViSioN! ?

Eiffel ENViSioN! is Eiffel Software's Eiffel-language plug-in for Microsoft's Visual Studio .NET™. It allows developers to use the Eiffel method and language seamlessly within the Visual Studio .NET development environment (until now, the ISE Eiffel compiler has only been available within EiffelStudio™, Eiffel Software's multi-platform IDE).

Seamlessly addressing the whole life cycle of software development, Eiffel ENViSioN! provides many of the facilities found in EiffelStudio that you can use to develop your application from requirements and modeling, through design, right through to time of deployment. Eiffel ENViSioN! also has superb browsing mechanisms for viewing information about your code and how that code performs and behaves whilst executing.

What is the difference between Eiffel, Eiffel ENViSioN! and EiffelStudio?

Eiffel is the *language* a developer uses to write great software. **EiffelStudio** is Eiffel Software's *multi-platform environment* and toolkit that surround the Eiffel language; it contains several productivity-related tools that are useful for creating large, sustainable, business-critical applications. **Eiffel ENViSioN!** is a **plug-in for Microsoft's Visual Studio .NET** development environment that allows users to use the Eiffel language at the same level as other .NET languages such as VB.NET and C#. (The Visual Studio .NET environment runs only on Windows on Microsoft's .NET Framework™, but has the capability to handle multiple languages.)

What does Eiffel ENViSioN! bring to Visual Studio .NET?

Eiffel ENViSioN! brings strengths in several areas that result in significant benefits to the developer:

- It is easy to learn
- More functions (can take advantage of all the features of .NET, plus more... see below)
- Greater reusability than objects created with any other language
- Can be used to increase the quality of existing .NET work
- Can be used to more easily create high-quality libraries for use across an enterprise
- Eiffel's "code once, run on many platforms" feature provides "future proofing" for companies using .NET: their users can migrate applications to and from different platforms, and can also use component libraries across platforms.

Eiffel brings quite a few unique and powerful features to .NET. Most noticeably it is the only .NET language to offer multiple inheritance and genericity. These two mechanisms (described in more detail in the Eiffel language FAQ) are an indispensable aid to creating fully reusable software in a truly object-oriented way.

Eiffel for .NET also brings the full benefits of Design By Contract to the software created, ensuring correctness of the software text, reliability and robustness. Whilst some other .NET languages do support contract mechanisms, Eiffel for .NET is the only one to support them natively, as an actual language construct. This is in accordance with classic Eiffel which considers contract use a vital and necessary element in the production of reliable, robust code.

Design by Contract can be used to improve the quality of existing .NET applications, by bringing the code in and adding contract assertions to it.

Eiffel also brings some existing libraries to .NET, so aside from being able to use other .NET libraries such as Windows.Forms for graphical elements you could instead use Eiffel's WEL or EiffelVision2 libraries. EiffelVision2 is a particularly unique graphical library in that it is multi-platform; thereby ensuring the compiled system will produce the same display and behavior on all supported platforms.

How is Eiffel ENViSioN! different from other .NET languages?

Eiffel is widely regarded as one of the easiest programming languages to learn. This was intentional - the idea being to remove syntax from the process as much as possible, so that the developer is free to focus on the design itself (and not on deciphering the code).

On the other hand, Eiffel is among the most fully-featured and powerful programming languages. When asked what limitations they have been faced with, users tend to say things like “There are no limitations. I love Eiffel.” While the part about “no limitations” may not be truly achievable, there are fewer limitations than with almost any other language. And other Eiffel features (many of them unique to Eiffel) such as polymorphism, genericity, multiple inheritance (called “the Holy Grail of O-O by CoDe magazine) and especially native Design by Contract make the Eiffel design experience something more freeing (and arguably, more fun) than that of other languages.

Eiffel ENViSioN! has a superb browsing mechanism for viewing information about your code and how that code performs and behaves while executing.

Eiffel ENViSioN!'s fully featured debugger supports Eiffel's Design by Contract methodology, which not only minimizes bugs in the first place, but then helps locate any remaining bugs for you. This helps to minimize the huge cost of maintenance of systems designed with other languages. Coupled with a fully functional, browsable editor, Eiffel ENViSioN! allows you to navigate to any part of your system easily to track down and fix the so called 'hard to find' bugs, therefore reducing project costs even further.

What operating systems does Eiffel ENViSioN! run on?

While Eiffel and EiffelStudio are very portable and run on many platforms, Eiffel ENViSioN! is targeted specifically at .NET, and thus runs only on Windows versions that support the .NET framework. At this time, those are Windows NT, Windows 2000, and Windows XP.

Why the emphasis on reusability with Eiffel ENViSioN! ?

Reusability is a strong theme in all of Eiffel programming, because it helps to increase quality and productivity simultaneously. Much of programming, when you think about it, is doing very similar tasks over and over again. Imagine, then, what your life (and your productivity) would be like if you could just do a task ONCE, in a very robust and generic way, and then could reuse that object any time you did that or a similar task. You might not ever have to start a new project from scratch again!

Is Eiffel ENViSioN! compatible with other .NET languages?

Are the applications I create in Eiffel ENViSioN! compatible with the other applications I've created in other .NET languages? The answer is YES, they are. Eiffel generates to the .NET CLR and provides complete integration and interoperability with components developed in other .NET languages. ENViSioN! also features a contract wizard that can be used to add contracts to existing non-Eiffel .NET classes. (This is cool.)

Is Eiffel ENViSioN! compatible with EiffelStudio?

Yes. Any Eiffel application created in either is fully available to the other.

Who should use Eiffel ENViSioN! ?

Anyone interested in creating high quality, robust, reusable, or business-critical software applications will love programming in Eiffel ENViSioN!. ENViSioN! will be interesting to existing Visual Studio .NET users who want to increase the quality of their systems (or the systems of others created in VS.NET). It allows integration across languages, which will allow users to employ the power of Design by Contract and the Eiffel debugger to progressively increase the quality of components that their entire company uses.

Won't I have to forsake my existing software, thus losing millions of dollars?

Absolutely not!

Eiffel is an open system; it is at its best when used as a combination technology to reuse software components written in various languages. In particular, Eiffel includes a sophisticated C and C++ interface, supporting:

- Calling C functions from Eiffel.
- Accessing C++ classes and all their components (functions or "methods", data members, constructors, destructors etc.) from Eiffel.
- Accessing Eiffel mechanisms from C or C++ through the Cecil library (C-Eiffel Call-In Library).
- Automatically producing a "wrapper" Eiffel class from a C++ class.

Eiffel makes it possible to move to modern software technology while reusing the best results of earlier practices.

Are there performance issues with Eiffel for .NET?

In the end, no. Support for multiple inheritance and the inclusion of the base library classes currently does result in generated *assemblies* which are larger than those generated by other .NET languages; however, there are no significant performance differences at run-time.

Is Eiffel transparent in Visual Studio .NET?

Yes. Eiffel for .NET can be used in exactly the same way as classic Eiffel and no extra language constructs are required to produce applications. Since there are some elements of the .NET language which have no equivalents in Eiffel, such as the ability to define custom attributes, these have been extended to Eiffel for .NET so that the full features of .NET can be used.

What is BON?

BON is the acronym for Business Object Notation, an analysis and design method that is based on concepts close to those of Eiffel (seamlessness, reversibility, contracting) and defines simple, intuitive graphical conventions. BON is particularly notable for its ability to scale up when you need to describe large and complex systems, keeping a view of the whole while zooming into the details of components at various levels of abstraction. The BON method is described in a widely acclaimed book by the authors of the method. Eiffel Software's EiffelCase (now the Diagram Tool within Eiffel ENViSioN!) analysis and design workbench directly supports BON.

Eiffel ENViSioN! does not support UML tools, however EiffelStudio does support this functionality if needed.

About Eiffel Software

Eiffel Software (a division of ISE) is the world leader in Eiffel pure object-oriented programming tools. Founded in 1985, Eiffel Software produces proven professional tools and component libraries for business-critical and enterprise software developments. Eiffel Software's products enable their customers to output more and higher-quality software in less time than with any other development tools available. Its users span the globe, in industries ranging from large financial institutions, to technology manufacturing, to government and defense contractors, to health care providers and more.