ABSYNTAX

Exposing Software Components for Wider Reuse[1]
White Paper

What is an application?

Commonly, a computer application is an amalgam of software "libraries". A library often represents a reusable collection of what might be loosely termed "classes". Finally, a class is a template for a unit of functionality that is able to manipulate information ("state") and do things ("behaviour"). Different classes define different states and behaviours. In non-compiled form, classes are also containers for programming constructs: in other words, classes are the embodiment of programming skills. Much intellectual property is locked inside classes.

Software specialists develop applications by creating and/or reusing libraries of classes, then packaging these libraries such that the relevant classes can be arranged to work with one another. The resulting arrangement determines the capabilities of the application as a whole. Alternative arrangements yield different capabilities, so this is a fundamental step in application-building.

For consumers, an application is an atomic entity that offers nothing of its subatomic constituents. Other than via configuration, consumers generally have no control over the capabilities supported by an application; in other words, they are forced to accept the decisions made by the software specialists during the application's development.

Components

The concept of component-oriented application architecture has been with us for years. Simply put, a component is a class with a clear, well-understood purpose and which is capable of operating independently in a variety of contexts. Components are often born of an architectural desire to decouple and expose independent, versatile capabilities for widespread reuse.

Unfortunately, this "reuse" is limited to other components and applications; that is, entities created by software specialists. This is a shame because it means that computer users without the relevant skills are unable to harness the benefits of such components.

The question is, can we expose these components – indeed, any suitable library-bound classes – to consumers in a way that does not require software engineering expertise to make use of them?

[1] The author acknowledges that the descriptions herein, while prevalent in the discipline of software engineering, are by no means universal, and that computer applications may be designed and developed using any number of disparate methodologies and technologies. Some simplifications have been made for the sake of readability.

Introducing Absyntax

Absyntax is an industry-agnostic desktop framework application that bridges the gulf between the need for bespoke software and the ability to deliver it. It helps people without traditional programming skills to develop software that may be executed both stand-alone and via third-party applications, and at the same time sidestep the need to understand the myriad of jargon that accompanies the software industry. In other words, Absyntax is a platform on which ordinary computer users can combine *components*.

Developing software with Absyntax is a visual process based on a very simple paradigm:

- select the required components (known as "features");
- configure where necessary;
- combine these features using connections.

Out of the box, Absyntax defines many granular, reusable features. Users are free to select and combine these features in manifold ways, thus realising an unlimited variety of operations. In addition, third-party software specialists can extend the framework by developing new features or exposing selected classes and components of their own software libraries as features. In this way, common, industry- or client-specific tasks are made more readily consumable by Absyntax users.

Conclusion

If you are a class library or application developer, consider exposing key classes as Absyntax features. This is typically a straightforward task requiring no modification of your existing code base. Your clients will benefit from the ability to consume your intellectual property on their own terms, without the constraints and interoperability issues suffered by applications.

Download the free, fully capable trial version from www.absyntax.com.

Contact us at support@absyntax.com.