Coding Standards in the Real World

Ensuring that your team’s code conforms to an agreed standard is an essential part of successful software development. By implementing coding standards, you gain immediate and long term benefits.

Contents

Benefits of Coding Standards ........................................................................................................................................... 2
Problems Implementing Coding Standards ......................................................................................................................... 2
The recommended approach .................................................................................................................................................... 3
  Step 1  – Get business owner’s buy-in ................................................................................................................................. 4
  Step 2  – Get initial consensus ............................................................................................................................................. 4
  Step 3  – Choose a base standard to follow ......................................................................................................................... 4
  Step 3.a – Customize the standard (optional) ...................................................................................................................... 4
  Step 4  – Create your own team guidelines document ....................................................................................................... 4
  Step 4.a – Prioritize what’s most important .......................................................................................................................... 4
  Step 5  – Implement Code Reviews ....................................................................................................................................... 5
  Step 6  – Use Code Generation ............................................................................................................................................ 5
  Step 7  – Review status and give feedback ........................................................................................................................... 5
How CodeIt.Right helps ......................................................................................................................................................... 5
Summary .................................................................................................................................................................................. 9
Benefits of Coding Standards

One of the immediate benefits is that all members of the team can understand and work with code written by other team members while the project is being built. But beyond that, coding standards help to avoid common and time-consuming problems once the project has moved into the maintenance phase.

The software product must continue to be maintained throughout its lifecycle. Developers leave, are assigned to other projects, or simply forget the detail of their original work on the project. Without coding standards, the organization has to analyze and relearn what was originally done, or even duplicate the original coding to find and then correct the problem.

By automating the analysis and documentation of the code, the organization will always have a record of what was done and can be assured that the coding style is consistent across the project. As a result, it is possible to identify and fix problems faster and without duplicating previous efforts.

The business owner will quickly see the significance of reduced long term costs and better productivity. Team leaders will find it easier to carry out code reviews when the same style is used throughout the project. Developers will feel comfortable with working with any part of the code base, no matter who first created it or who has made any changes to it.

In summary, coding standards help reduce the long-term cost of the project and makes the best use of expensive developer resources.

Problems Implementing Coding Standards

People resist change. This applies to changes to an individual’s working practices as much as anything else. So the primary problem with implementing a coding standards system is overcoming that resistance. And the way to overcome that resistance is to help those involved see the benefits that are in it for them and to alleviate any potential fear of failure caused by the change.

A SubMain survey of developers and teams revealed that out of respondents who attempted to implemented guidelines using manual code reviews only 40% had succeeded. The poll highlighted the main problem areas.

Looking at the reasons for those failed implementations, it’s clear that most of them (84%) are because the developers couldn’t agree on or follow guidelines –

- Couldn’t get a consensus on which standard to follow
- Resistance among the team members
- Developers forgetting to keep to the guidelines

<table>
<thead>
<tr>
<th>Coding Standards Benefits</th>
</tr>
</thead>
<tbody>
<tr>
<td>✓ Code Clarity/Easier to Understand</td>
</tr>
<tr>
<td>✓ Easier to Maintain</td>
</tr>
<tr>
<td>✓ Reduces Bugs</td>
</tr>
<tr>
<td>✓ Simplifies Code Reviews</td>
</tr>
<tr>
<td>✓ Shorter learning curve for new team members</td>
</tr>
<tr>
<td>✓ Consistency across large and distributed teams</td>
</tr>
<tr>
<td>✓ Comply with internal or regulatory quality initiatives</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Business Benefits</th>
</tr>
</thead>
<tbody>
<tr>
<td>✓ Improve software quality</td>
</tr>
<tr>
<td>✓ Accelerate time to market</td>
</tr>
<tr>
<td>✓ Enhance customer satisfaction</td>
</tr>
<tr>
<td>✓ Reduce long term cost</td>
</tr>
<tr>
<td>✓ Improve productivity</td>
</tr>
</tbody>
</table>
And even in a further 10% of failures, management thought implementing and running a manual review system was too expensive.

Using an automated tool to implement team guidelines overcomes all these major problem areas. If developers keep forgetting, a tool will help by automatically fixing the issue or reminding the developer of what’s needed.

Debates about which standard to follow are avoided because the tool comes with a base standard which can then be easily customized and fine-tuned to meet team requirements.

A tool relieves the developer of the bulk of the tedious tasks of analyzing, reviewing and correcting code that doesn’t meet the guidelines.

In all cases the use of an automated tool overcomes the problem areas highlighted by the poll.

One major disadvantage of manual reviews is the length of time it takes to get feedback.

Another problem is that emotional influences can negatively affect the process. Developers being reviewed often take comments and suggestion for changes that are made during peer reviews personally as criticism. Unfortunately there is no simple solution to this very human issue. Using an automated review system as the first and main part of the process avoids these problems.

### The recommended approach

In an ideal world, everyone would see the benefits and be willing to change. In the real world, it’s not that easy. So, for this to succeed, everyone involved has to buy in to the process at some level. This must start with the business owner and flow down through team leaders to the developer team.

The only sure way to get all the developers on board is to include them in the discussions. Experienced developers may find it hard to break old habits and will have strong views on what works and what doesn’t. Getting their input and achieving a consensus is crucial to a successful implementation. Not only is it important for them to be personally committed to the scheme, their status influences junior developers. If the senior members are negative, this negativity will filter down to the junior members.
**Step 1 - Get the business owner’s buy-in**
The business owner needs to know the benefits of following a standard, as well as the costs that will be incurred. This is particularly important for manual code reviews, where the costs of staff time will usually be very high.

**Step 2 - Get initial consensus**
Involve all affected parties and analyze the options, discuss the possibilities, achieve an initial consensus.

The goal is to reach the best consensus you can. This will be foundation that can be built on or improved later, once a system is actually in place. Therefore it doesn’t have to be the ideal solution; it just has to be a solution that makes a start along the road.

**Step 3 - Choose a base standard to follow**
In the discussions, don’t get sucked into curly brace wars, and stay focused on the more important aspects of the standards. Look at the most common standards and discuss the pros and cons. Choose a base standard to follow, preferably the one that has the support of the majority. Keep in mind that any standard is better than no standard at all.

The most popular standards are shown in the sidebar on the right. Our research has shown that across the industry Microsoft Design Guidelines are the most used and SubMain customers mostly choose those or the SubMain .NET Coding Guidelines. Of the companies we have surveyed, only a small number use IDesign or Philips C# Coding Standards.

**Step 3.a - Customize the standard (optional)**
Once a standard has been agreed, it can be customized to fit your team needs and requirements. However, using common industry practices makes it easier for your developers to follow and for new developers to fit in, so you should try to avoid this unless you absolutely have to.

**Some Coding Standards**
- Microsoft Design Guidelines for Developing Class Libraries  
  [http://submain.com/fwlink/std/ms](http://submain.com/fwlink/std/ms)
- SubMain .NET Coding Guidelines – MS Guidelines extended and adopted for both VB and C#  
  [http://submain.com/fwlink/std/sm](http://submain.com焌/lendir/std/sm)
- David McCarter’s .NET Coding Standards  
  [http://submain.com/fwlink/std/dm](http://submain.com焌/lendir/std/dm)
- IDesign C# Coding Standards
- Philips C# Coding Standards

**Step 4 - Create your own team guidelines document**
Create a document that reflects the consensus reached in Step 2. Be prepared to amend this document in the light of experience once coding standards are in use.

**Step 4.a - Prioritize what’s most important**
Split the document into ‘Required’ and ‘Recommended’ sections. Keep the ‘Required’ section as small as reasonably possible.

Bear in mind that change can be hard to implement, so don’t include too many minor items that make very little difference to the overall quality.
Step 5 - Implement Code Reviews
Implement a Code Review system. This is best carried out as a combination of manual and automated reviews.

Using an automated code review tool such as CodeIt.Right, followed by a traditional manual code review is the best approach. The tool will find things that humans will miss, whether through lack of concentration, lack of knowledge, misunderstandings or any other human failing. Then the human code review session can focus on the proprietary business logic and ensure that it is implemented correctly.

A major hurdle when implementing code reviews is that some developers take it as a personal criticism when code smells are flagged up in their code. It’s important for them to be reassured that this isn’t the case and to see that code creation is a team effort and the team’s total output must have a consistent structure. Consistency is more important than personal style. If everyone is working to the same standard, they will feel comfortable in any part of the code base. As a result, you have a more flexible team in which you can easily move people around, have a developer seamlessly take over another’s code, and quickly bring new team members up to speed.

Step 6 - Use code generation
Not so much a step, but rather this is a recommendation of good practice. Code generation is the process of using a tool or code to write code that would otherwise have to be written by a person. This offers many benefits including saving developers’ time, ensuring consistency and guaranteeing accuracy. By removing the repetitive basic code creation chores, your developers are free to work on tasks that require their individual skill. The code generation tool can be configured to ensure that your chosen standards are always adhered to in the generated code. Tools such as the CodeSmith Generator – [http://submain.com/fwlink/codegen/tool](http://submain.com/fwlink/codegen/tool) - can be used, or you can create your own using T4 Templates – [http://submain.com/fwlink/codegen/t4](http://submain.com/fwlink/codegen/t4) and [http://submain.com/fwlink/codegen/kd](http://submain.com/fwlink/codegen/kd). There is also a number of books and articles on code generation in .NET - [http://submain.com/fwlink/codegen/books](http://submain.com/fwlink/codegen/books).

Step 7 - Review status and give feedback
Review the initial agreement. If necessary, make changes to the guidelines document.

Track progress and report back to the business owner. To avoid a report that is too subjective, you need reportable metrics that monitor progress and identify problem areas.

How CodeIt.Right helps
CodeIt.Right is a tool that will help the successful implementation of each step of the process described above.

Get the business owner's buy-in
The business owner needs a clear picture of the benefits and costs. The savings when using CodeIt.Right to automate a code review process that would traditionally be performed manually can be calculated up front. These cost savings and the ROI (return on investment) will be key factors in the business owner’s decision. Knowing the clear benefits of the combined Automated/Manual approach will help to bring the business owner on board. You can access CodeIt.Right ROI calculator here - [http://submain.com/fwlink/cir-roi](http://submain.com/fwlink/cir-roi)
Get initial consensus
It’s easier to get a consensus when you have a tool that offers you guidelines out of the box. This means that you don’t have long arguments about what guidelines should be used. In this way, CodeIt.Right gives you a head start on this often very difficult step.

Choose a base standard to follow
CodeIt.Right already offers the most popular standard in the industry - the Microsoft Coding Guidelines. In addition, the SubMain .NET Coding Guidelines document, complementary with CodeIt.Right, offer expanded and improved version of the Microsoft Design Guidelines.

Customize the standard
If you want to use any of the alternative standards, you can easily customize CodeIt.Right for that standard to be the base rule set to fit your needs. Here is the list of some of the CodeIt.Right techniques you can use for this

- custom profiles
- rules instances
- rule overrides
- custom rules

Create your own team guidelines document
Once you’ve chosen the base standard and customized it to your needs, you can use CodeIt.Right to generate your guidelines document automatically. It does this based on the profiles and rules you have configured and so will create a document which precisely reproduces your customized guidelines. The document can be viewed on screen or printed out, so you can have a Company Guidelines Manual that can be used for reference. Everything in the printed manual is included in the guidelines and rules that will be enforced when CodeIt.Right runs.

Prioritize what’s most important
CodeIt.Right offers many features that allow the use of a drilldown approach to coding guidelines and automated reviews. This is particularly useful when working with an existing code base that was written before the team coding standards were introduced. You can create any number of different profiles based on criteria of your choice. So you can create, for example, a ‘Required’ profile. This profile would be filtered to include the most important rules. In the same way, you could create a ‘Recommended’ profile in which the rules would deal with less important validations. You can in fact create further profiles that deal with specific areas - a security rules profile or a performance rules profile, for instance.

Within each profile, the individual rules can be tweaked in many ways. Firstly, if you base a profile on a standard set of guidelines, such as the Microsoft Design Guidelines, you can choose to include or exclude any of its rules according to your requirements. Then, you can also set or alter the Severity of the violation, or the scope, target, or the configurable rule properties.
The Severity Threshold in the OnDemand mode and the Command Line version allows for the temporary filtering out of issues that don’t meet a specific Severity Threshold. For example, the initial review can be limited to reporting only one or more of the higher categories, such as ‘Error’. At this stage, lower severity issues can be temporarily ignored. As a result, the number of violations that need to be addressed at this stage will be less overwhelming and more manageable. Of course, at a later stage, the filtering of the Severity Threshold can be changed again, allowing for lower severity issues to be flagged up and fixed.

CodeIt.Right also allows you to exclude projects, files and specific violations.
Implement Code Reviews

CodeIt.Right allows for large scale automation of the code review process. The most efficient in terms of finding issues early and fixing them is the Instant Code Review. This carries out checks and validations in the background and provides immediate feedback to the developer so that problems can be solved on the spot. This feature, combined with automated refactoring, means that it finds the issues and helps to fix them immediately.

The cost of fixing issues increases exponentially as the development process continues. As can be seen from the diagram below, see Figure 5, the cost of fixing an error in the early design, coding and unit test stages is relatively low. The light blue rectangle in the diagram highlights the fact that because most errors are obviously introduced in the coding and test stages, fixing those errors early is the most cost effective approach.

The yellow line on the chart shows that most errors are discovered in the later stages of the process. To avoid the increased costs that this causes, the best policy is to employ methods such as CodeIt.Right Instant Review that guarantees that coding standards are followed in the initial coding stage.

An alternative approach, using OnDemand reviews with CodeIt.Right allows for a specific set of rules to be run on an as-needed basis. A detailed violations report is generated, offering options to fix the errors individually or in bulk. See Figure 3 for sample screenshot.

Check-In Policy Integration allows for code to be reviewed automatically before it is checked in to source control. This ensures that source control then only contains clean code.

The command line or automated build version allows the reviews to be carried out during the build. The reports generated during this process can then be distributed to whoever needs them.
Use code generation
The Refactoring to Patterns feature of CodeIt.Right is similar to code generation because it implements correct patterns, finds inconsistencies and fixes broken patterns. In the same way as generated code, Codelt.Right’s code output is correct and conformant.

Codelt.Right works well with code generation because it allows you to set options that will exclude the checking of generated code when marked with [ComplierGeneratedAttribute] or [GeneratedCodeAttribute]

You can also have Codelt.Right ignore standard and your custom regions that wrap around auto-generated code.

Review status and give feedback
Codelt.Right comes with a wide range of analytical tools that include violation reports, code metrics reports and very flexible pivot reporting tools. Not only will these tools create reports in pre-set formats, you can build your own reports configured precisely to your requirements. These can be used to track and report progress back to the business owner. Reports can be printed, exported or emailed to management, providing quantitative analysis and benchmarking. The Statistics feature in the Enterprise Edition keeps track of the health of the project over time and can be used to highlight trends.

When reviewing the project status and providing feedback, it is important to use data that is objective. The reports available in Codelt.Right include the metrics needed to accurately and objectively monitor progress and identify problem areas.

Summary
While it is very important to follow a coding standard and best industry practices, it isn’t always easy or straightforward to do so. There are major long term benefits to be gained, but the hurdles of additional cost and human resistance to change must first be overcome. By following the approach recommended in this white paper you will be able to get your team to implement and use an agreed coding standard with the least amount of conflict. The advice contained here is based on many lessons learned from failed implementations as well as many success stories.

Using Codelt.Right to help you complete all the steps described will make the implementation easier, less confrontational and faster. The quality of code output will be improved, you will be conforming to accepted team and industry standards, and all this will be achieved at less cost.

For information on SubMain’s Codelt.Right coding standards and automated code review product contact us at customer-service@submain.com
About SubMain

Thousands of customers use SubMain software quality and developer productivity tools to ensure the high quality and performance of their desktop, mobile, Web and cloud-based applications. SubMain products are easy to use and deploy, are affordable and available for trial at the website. Learn more about SubMain, the company's products or join the active user community at http://submain.com.