Lars Hvam
May 11, 2018 1 minute read

All your ABAP prefixes are belong to us

Prefixes

Prefixing variables in ABAP is a hot topic, and during the years there have been many interesting discussions:
https://blogs.sap.com/2018/04/30/are-30-characters-enough-to-make-your-code-better/
https://blogs.sap.com/2016/02/05/fanning-the-flames-prefixing-variableattribute-names/
https://blogs.sap.com/2009/08/30/nomen-est-omen-abap-naming-conventions/
I find it most important that naming is done in a consistent way, all using prefixes or no prefixes at all. However, all customers I have worked for recently have prefixing as part of their development guidelines, typically supported by the "Extended Naming Convention for Programs" in Code Inspector/ATC. This blog introduces a new Code Inspector check which can be used for making sure prefixes follow the development guidelines.

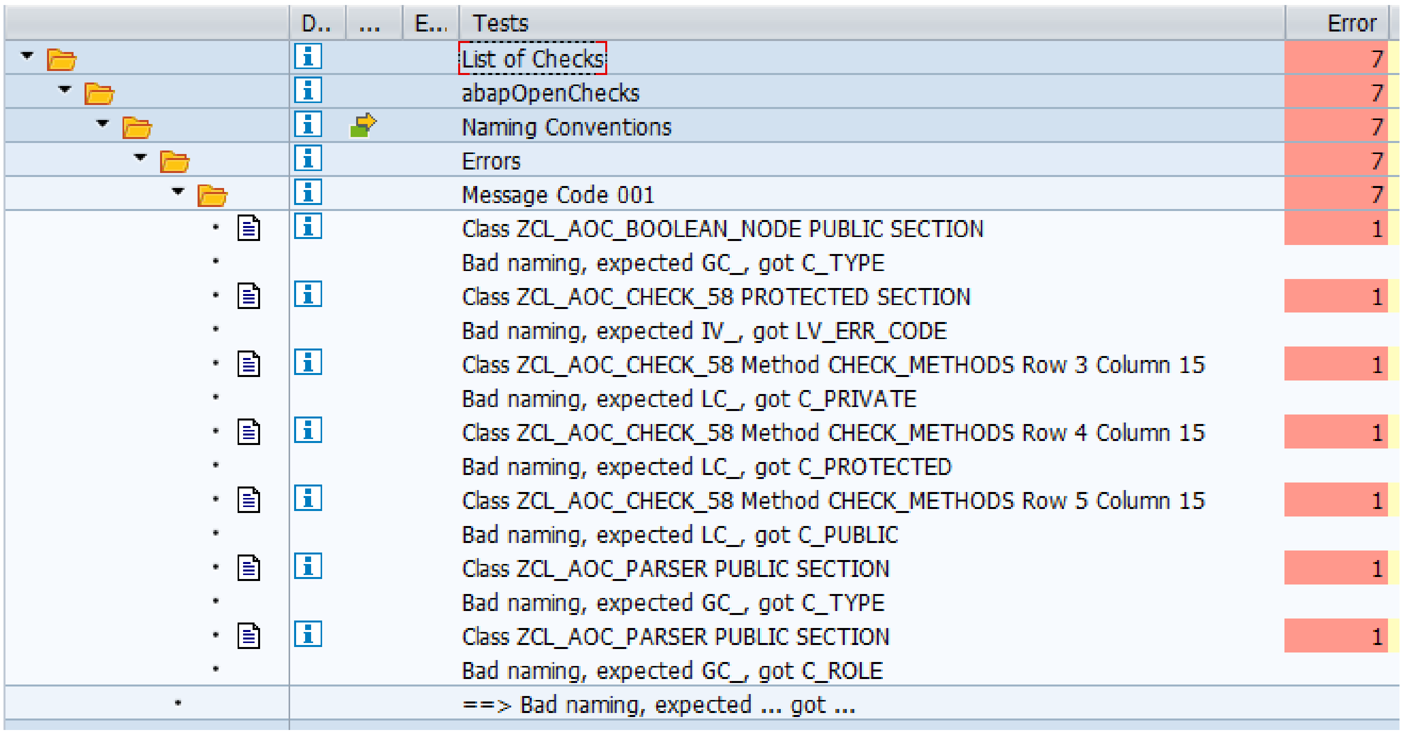
abapOpenChecks

abapOpenSource is an open source project which provides extra checks for Code Inspector/ATC, a lot has happened since it was initially released 3 years ago. The number of checks has increased from 26 to 69, and a lot of false positives been removed from the results. All checks have different use cases and vary in quality. I recommend trying it out, the checks are widely configurable, and it is possible to cherry-pick the most useful checks.

All your prefixes

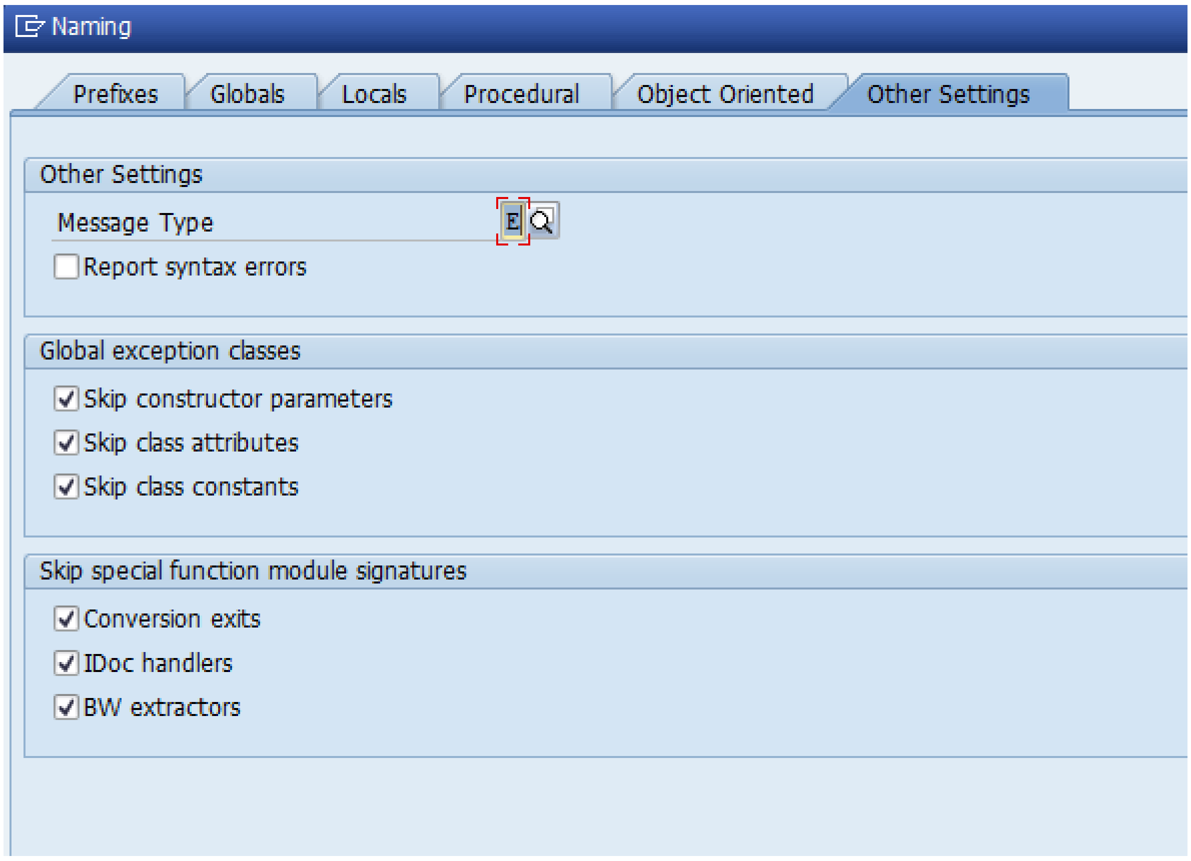
The latest check "Naming Conventions" can check prefixes for variables in ABAP and report the findings to the developer.

When reporting findings, the check reports the expected prefix so the developer will not have to look up the conventions or the type of the variable,



	D..	...	E...	Tests	Error
	ℹ			List of Checks	7
	ℹ			abapOpenChecks	7
	ℹ			Naming Conventions	7
	ℹ			Errors	7
	ℹ			Message Code 001	7
	ℹ			Class ZCL_AOC_BOOLEAN_NODE PUBLIC SECTION	1
				Bad naming, expected GC_, got C_TYPE	
	ℹ			Class ZCL_AOC_CHECK_58 PROTECTED SECTION	1
				Bad naming, expected IV_, got LV_ERR_CODE	
	ℹ			Class ZCL_AOC_CHECK_58 Method CHECK_METHODS Row 3 Column 15	1
				Bad naming, expected LC_, got C_PRIVATE	
	ℹ			Class ZCL_AOC_CHECK_58 Method CHECK_METHODS Row 4 Column 15	1
				Bad naming, expected LC_, got C_PROTECTED	
	ℹ			Class ZCL_AOC_CHECK_58 Method CHECK_METHODS Row 5 Column 15	1
				Bad naming, expected LC_, got C_PUBLIC	
	ℹ			Class ZCL_AOC_PARSER PUBLIC SECTION	1
				Bad naming, expected GC_, got C_TYPE	
	ℹ			Class ZCL_AOC_PARSER PUBLIC SECTION	1
				Bad naming, expected GC_, got C_ROLE	
				==> Bad naming, expected ... got ...	

As part of the check configuration, it will by default ignore most of the naming in exception classes, as it sometimes is not possible to have these follow the conventions. Also, special function module signatures are ignored, like conversion exits where the parameters are always INPUT and OUTPUT,



Other Settings

Message Type: E

☐ Report syntax errors

Global exception classes

☑ Skip constructor parameters

☑ Skip class attributes

☑ Skip class constants

Skip special function module signatures

☑ Conversion exits

☑ IDoc handlers

☑ BW extractors

Overall it should report less false positives, and if there are false positives, it is possible to inherit from the check class and implement custom rules or give back to the community via a PR on GitHub.

As always, bug reports and suggestions welcome

Alert Moderator

Assigned tags

ABAP Development | abap | abap test cockpit | abapopenchecks | atc |

View more...

Related Blog Posts

Using Categories in the ABAP development space
By **Former Member** , Feb 10, 2015

All Consultants are Evil doers meant to KILL your system
By **Former Member** , Jul 20, 2015

abap2xlsx – I'm no longer a baby, I grow up and now I'm a fully featured tool
By **Ivan Femia** , Nov 25, 2010

Related Questions

prefixes ABAP functions
By **Former Member** , Apr 14, 2007

From which table could i get all the Variants of a specific ABAP program?
By **Former Member** , Aug 15, 2007

Is there naming conventions for variables, itabs?
By **Ricardo Jr. Caliolio** , Jan 30, 2008

16 Comments

Mike Pokraka

May 11, 2018 at 11:15 am

"*However, all customers I have worked for recently have prefixing as part of their development guidelines*"

That's because most guidelines originate from last century when prefixing made sense. Apparently change is hard ☺

Great blog, useful tools, as always!

Liked (5) Reply Alert Moderator

Jakub Filak

May 11, 2018 at 12:52 pm

IMHO, variable prefixes (aka Hungarian notation) makes refactoring more painful and results into unnecessary noise in git commits.

Like (1) Reply Alert Moderator

Lars Hvam | Post author

May 11, 2018 at 12:56 pm

yeah, but if the customer requires prefixes, I hope to make it a bit easier for the developer to align the code to guidelines after refactoring

Like (0) Reply Edit Alert Moderator Trash

Chris Paine

May 14, 2018 at 7:56 am

walking along coastal path spots a nice beach decides to go investigate along cliff edge if there is a safe way down… Along walks a very friendly fellow who offers me a parachute.

Me: “ummm thanks, but I think I’ll just look for a path down”

Parachute person: “but we always parachute down to this beach”

Me : “has anyone ever tried walking down?”

Parachute person: “but it’s not allowed!”

Me: “why?”

Parachute person: hands over 90 page beach exploring manual printed in 1993.

Me: “but this is over 20 years old! Why are you doing this?”

Parachute person: “hey, it’s not me you have to ask mate! You’ll have to speak to the technical architects!”

Me: “sigh, ok gimme the darn parachute…”

And a very nice parachute it was! 😉

Like (2) Reply Alert Moderator

Jelena Perfiljeva

May 14, 2018 at 9:15 pm

This is very timely as just this morning I got an email with the note that I need to add prefix “C_” to these constants in my program:

```
CONSTANTS: delivery TYPE vbtyp VALUE 'J',
           order_display_transaction TYPE sy-tcode VALUE 'VA03',
           replenishment_group TYPE lvc_spgrp VALUE 'REPL',
           customer_group TYPE lvc_spgrp VALUE 'CUST'.
```

This is according to the internal “ABAP guidelines” that don’t even exist in any documented form and I’m guessing are just passed from generation to generation as some kind of a folks tale.

I’m still having hard time parting with my LV_s and ITAB_s. Once I tried to use the descriptive name like “orders” but then it did not go well because apparently SAP already got dibs on it. 🙂 Fool me once.

But when we already have constants like “space” or “abap_true” what’s the point prefixing our own?

DSAG ABAP guidelines moved on as well:

> *The first version of these guidelines supported a long naming convention variant, which on the basis of our own experiences and the current position of debate in trade literature and the SCN (see Appendix A.3) we no longer advocate.*

P.S. There was another epic blog on the subject:
https://blogs.sap.com/2015/09/22/hungarian-beginners-course-a-polemic-scripture-against-hungarian-notation/

Like (2) Reply Alert Moderator

Matthew Billingham

May 15, 2018 at 5:24 am

Not just DSAG, but help.sap.com. One of my clients recently updated their standards with the usual **lv_ lt_** nonsense (who, me , biased?). I posted on an internal forum that this is inconsistent with DSAG’s latest recommendations and SAP’s own guidelines. I added some explanation as to why so many prefixes are simply not needed and may even hinder readability.

The standards “owner” and his team responded, and, after I showed them one of my programs without certain prefixes, they’re changing the requirements.

Some people, when you talk about prefixes, think you're abandoning all of them. This is not the case. It's only about getting rid of indicators of variable type, not of use. Nowadays, there should be very very few global variables, so it makes sense to state that a non-prefixed variable is local. But global variables do need to be indicated as such.

I see your point about constants, but I'll continue to use **c_**. I think it's useful to know that a constant at sight. SPACE is part of the abap language, but abap_false and abap_true are pretty much there as well. Hmm… you could really obfusticate a program by defining your own abap_false to be 'X' and abap_true to be ' '.

Like (5) **Reply** **Alert Moderator**

Lukas Weigelt

May 15, 2018 at 3:33 pm

I learned to cherish supposedly superfluous prefixes like lv_ lt_ and so on whilst maintaining spaghetti-code from the dark ages. One thing I never understood, though, is, why a lot of people use the prefix <fs_ for field-symbols… Has the syntax ever been different than nowadays, i.e. did field-symbols not have angle brackets 50 years ago? °-°

> *you could really obfusticate a program by defining your own abap_false to be 'X' and abap_true to be ' '.*

You evil genious, you 😛

Like (0) **Reply** **Alert Moderator**

Matthew Billingham

May 15, 2018 at 3:44 pm

I have a theory about <fs_. It also explains why some people use f_ to prefix their Form names. However, it might offend some practitioners so I won't share it. Suffice to say it may be connected to the question, 'Do you still have the boxes the computer came in?'

Like (1) **Reply** **Alert Moderator**

Jelena Perfiljeva

May 15, 2018 at 8:18 pm

Forget prefix, I've seen quite a few of <fs> names, literally. I suspect those were just copy-pasted from Help examples or from the interwebs. I might have done the same myself but only the first few times I've ever used field symbols many years ago. Much to my relief, that system no longer exists, so no one can see that horror. 🙂

Like (1) Reply Alert Moderator

Jelena Perfiljeva

May 15, 2018 at 8:28 pm

I believe even DSAG guidelines still suggest to use S_ and P_ for selection options and parameters. And there is I_, E_ for import/export and such. There are times when a prefix makes sense and is useful.

Although I'm in favor of keeping this as a guideline and not enforcing strictly as a standard.

Like (0) Reply Alert Moderator

Jürgen L

May 15, 2018 at 8:35 pm

they do

Like (0) Reply Alert Moderator

Matthew Billingham

May 16, 2018 at 5:23 am

I refer you to my previous comment.

> *Some people, when you talk about prefixes, think you're abandoning all of them. This is not the case. It's only about getting rid of indicators of variable type, not of use.*

S_, P_, I and E_ etc. are indicators of use. I'm a fan of those. They're also special cases of global variables. Global variables should have a g_ prefix. Then hunted down and destroyed as much as possible and never refered to within, e.g. local methods.

Like (1) Reply Alert Moderator

Michal Biegun

June 6, 2018 at 3:25 pm

Dear Jelena.

In blog you linked Ralf reasons, that prefixes indicating usage should stay. So Importing is a read only parameter therefore i_ is meaningful. I think that constant is the same case. I've switched to non-prefixing and meaningful naming, but I still struggle with c_. When I see c_ (or co_, or whatever) in code, I immediately know, that this element will not change the logic of code, no matter what happened before (because it's a constant). It's very important. Equally important is to know, that i_var (importing) comes from the outside of method. It tells a lot about logic.

Like (0)   Reply   Alert Moderator

Jelena Perfiljeva

June 7, 2018 at 8:26 pm

Regarding constants, see my recent submission to ABAP Gore thread. If we only look at the prefix, we don't "know", we just "assume". ☺

Like (1)   Reply   Alert Moderator

Michal Biegun

June 22, 2018 at 3:03 pm

If we look at name of function/method, we just assume it does what it says.

Many times I've seen things you've posted. I also saw methods/functions that do things exactly opposite to what their names suggest :/. There is really no efficient tool against sloppy developers (electric chair maybe..).

I could assume that constant is a constant and r_sth is a returning parameter, or I can just check it. Once checked, I don't have to remember which element was a variable and which was constant. Therefore if code is longer than 30 lines, I still find it useful to have constants distinguished.

Like (0)   Reply   Alert Moderator

Uwe Fetzer

May 15, 2018 at 9:26 am

Lucky me. I'm responsible for the guidelines and Code Inspector Checks (ATC) in my company ☺

Like (5)   Reply   Alert Moderator

Add Comment

Find us on