↖ Topic Index

🚧 🚦 Indentation Styles for C-like Programming Languages🚧

Indentation Style	while loop	for loop with nested if/else	Other example	cc-mode implementation properties	cperl-mode implementation properties
	The purpose of this table is to describe all aspects of the various indentation styles used by C-like programming languages (C, C++, D, Java, Perl, …) and identify the properties that are required to develop indentation style formatters that can generate these formats properly. The table lists the various indentation styles and provides code snippet examples that can then be used to identify all required properties.				

Allman	`while (x == y)`
	`{`
	`  foo();`
	`  bar();`
	`}`
		`int`
		`some_func()`
		`{`
		`  int i;`
		`  for (i = 0; i < 10; i++)`
		`  {`
		`  if (i % 2 == 0)`
		`  {`
		`  do_something(i);`
		`  }`
		`  else`
		`  {`
		`  do_something_else(i);`
		`  do_third_thing(i);`
		`  }`
		`  }`
		`  return i;`
		`}`
			`int`
			`some_func()`
			`{`
			`  int c;`
			`#ifdef HAS_GETCH`
			`  while ((c = getch()) != EOF)`
			`#else`
			`  while ((c = getchar()) != EOF)`
			`#endif`
			`  {`
			`  do_something(c);`
			`  }`
			`}`		

GNU
• **GNU-style code formatting**
	`while (x == y)`
	`  {`
	`  foo();`
	`  bar();`
	`  }`
		`int`
		`some_func()`
		`{`
		`  int i;`
		`  for (i = 0; i < 10; i++)`
		`  {`
		`  if (i % 2 == 0)`
		`  {`
		`  do_something(i);`
		`  }`
		`  else`
		`  {`
		`  do_something_else(i);`
		`  do_third_thing(i);`
		`  }`
		`  }`
		`  return i;`
		`}`			

Indentation Style	while loop	for loop with nested if/else	Other example	cc-mode implementation properties	cperl-mode implementation properties

Whitesmith	`while (x == y)`
	`    {`
	`    foo();`
	`    bar();`
	`    }`
		`int`
		`some_func()`
		`    {`
		`    int i;`
		`    for (i = 0; i < 10; i++)`
		`        {`
		`        if (i % 2 == 0)`
		`            {`
		`            do_something(i);`
		`            }`
		`        else`
		`            {`
		`            do_something_else(i);`
		`            do_third_thing(i);`
		`            }`
		`        }`
		`    return i;`
		`    }`			

K&R	`while (x == y) {`
	`    foo();`
	`    bar();`
	`}`
		`int`
		`some_func()`
		`{`
		`    int i;`
		`    for (i = 0; i < 10; i++) {`
		`        if (i % 2 == 0) {`
		`            do_something(i);`
		`        }`
		`        else {`
		`            do_something_else(i);`
		`            do_third_thing(i);`
		`        }`
		`    }`
		`    return i;`
		`}`			

K&R 1TB
One True Brace
	`while (x == y) {`
	`    foo();`
	`    bar();`
	`}`
		`int`
		`some_func() {`
		`    int i;`
		`    for (i = 0; i < 10; i++) {`
		`        if (i % 2 == 0) {`
		`            do_something(i);`
		`        }`
		`        else {`
		`            do_something_else(i);`
		`            do_third_thing(i);`
		`        }`
		`    }`
		`    return i;`
		`}`			

Indentation Style	while loop	for loop with nested if/else	Other example	cc-mode implementation properties	cperl-mode implementation properties

K&R Linux Kernel
• use hard tab characters
• tab stop every 8 spaces
• else is cuddled
		`int`
		`some_func() {`
		`⇥int i;`
		`⇥for (i = 0; i < 10; i++) {`
		`⇥⇥if (i % 2 == 0) {`
		`⇥⇥⇥do_something(i);`
		`⇥⇥} else {`
		`⇥⇥⇥do_something_else(i);`
		`⇥⇥⇥do_third_thing(i);`
		`⇥⇥}`
		`⇥}`
		`⇥return i;`
		`}`			

Ratliff
• Like K&R 1TB but the closing brace lines up with the indentation of the nested block.
	`while (x == y) {`
	`    foo();`
	`    bar();`
	`    }`
		`int`
		`some_func() {`
		`    int i;`
		`    for (i = 0; i < 10; i++) {`
		`        if (i % 2 == 0) {`
		`            do_something(i);`
		`            }`
		`        else {`
		`            do_something_else(i);`
		`            do_third_thing(i);`
		`            }`
		`        }`
		`    return i;`
		`    }`
			`for (i = 0; i < 10; i++) {`
			`    if (i % 2 == 0) {`
			`        do_something(i);`
			`        }`
			`    else {`
			`        do_something_else(i);`
			`        }`
			`    }`		

Horstmann
• Adapts Allman by placing the first statement of the block one the same line as the opening brace, reducing the. number of lines.
	`while (x == y)`
	`{   foo();`
	`    bar();`
	`}`
		`int`
		`some_func()`
		`{   int i;`
		`    for (i = 0; i < 10; i++)`
		`    {   if (i % 2 == 0)`
		`        {   do_something(i);`
		`        }`
		`        else`
		`        {   do_something_else(i);`
		`            do_third_thing(i);`
		`        }`
		`    }`
		`    return i;`
		`}`
			`while (x == y)`
			`{   something();`
			`    something_else();`
			`    //...`
			`    if (x < 0)`
			`    {   printf("Negative");`
			`        negative(x);`
			`    }`
			`    else`
			`    {   printf("Non-negative");`
			`        nonnegative(x);`
			`    }`
			`}`
			`final_thing();`		

Indentation Style	while loop	for loop with nested if/else	Other example	cc-mode implementation properties	cperl-mode implementation properties

Pico
• This is a style for the Pico language which lacks return statement.
• For C like languages it is similar to Horstmann, but the closing brace is at the end of the last indented statement, making it similar to the Lisp style below.
	`while (x == y)`
	`{  foo();`
	`  bar(); }`
		`int`
		`some_func()`
		`{  int i;`
		`  for (i = 0; i < 10; i++)`
		`  {  if (i % 2 == 0)`
		`  {  do_something(i); }`
		`  else`
		`  {  do_something_else(i);`
		`  do_third_thing(i); }}`
		`  return i; }`
			`stuff(n):`
			`{ x: 3 * n;`
			`  y: do_stuff(x);`
			`  y + x }`		

Lisp	`while (x == y)`
	`  { foo();`
	`  bar(); }`
		`int`
		`some_func()`
		`  { int i;`
		`  for (i = 0; i < 10; i++)`
		`  { if (i % 2 == 0)`
		`  { do_something(i); }`
		`  else`
		`  { do_something_else(i);`
		`  do_third_thing(i); }}`
		`  return i; }`
			`for (i = 0; i < 10; i++)`
			`  {if (i % 2 == 0)`
			`  {do_something(i);}`
			`  else`
			`  {do_something_else(i);`
			`  do_third_thing(i);}}`		

Cperl 🚧					
C++ 🚧 **(cperl C++ style)**					
PerlStyle 🚧					

PBP (Perl Best Practice) 🚧	`while (x == y) {`
	`  foo();`
	`  bar();`
	`}`
		`int`
		`some_func()`
		`{`
		`  int i;`
		`  for (i = 0; i < 10; i++) {`
		`  if (i % 2 == 0) {`
		`  do_something(i);`
		`  }`
		`  else {`
		`  do_something_else(i);`
		`  do_third_thing(i);`
		`  }`
		`  }`
		`  return i;`
		`}`			