

Forum:[Hack Forums](#)

Thread: [Difference between a while loop and a do while loop?](#)

hackaquack - July 30, 2014, 2:24 pm

I see no difference between a do while loop and a while loop.

A do while loop is basically a while loop with the loop body being declared before the expression.

What's the difference between

Code do {

statements

} while(expression);And

Code while(expression) {

statements

}

?

How can I make use of the do while loop?

Thanks,

- John

?XilluX - July 30, 2014, 2:38 pm

The expression gets checked on the while loop first and then the code will be executed. On the do while the code gets executed before the expression gets checked.

Labeler - July 30, 2014, 2:40 pm

While loops are good if you want to keep looping until something happens.

For loop is good to use if you want to loop a certain amount of times.

Do-while loops

These are the black swans of programming—they show up every once in a long while. The only real reason to use a do-while loop is if you want to do something at least once. A good example is the earlier sample code that prompts a user for a password, or, more generally, any kind of user interface that requires input and repeatedly presents a prompt to the user until the input is correct. In some cases, even if you do want the body of a loop to be repeated, it still might not be the best choice if the body needs to be slightly different the first time through—for example, if you want to have a different message if the user entered the wrong password.

For example, how would you write something like this with a do-while loop?

```
string password;
```

```
cout << "Enter your password: ";
```

```
cin >> password;
```

```
while ( password != "xyzy" )
```

```
{
```

```
cout << "Wrong password--try again: ";
cin >> password;
}
```

Do While loop:

```
string password;
do
{
if ( password == "" )
{
cout << "Enter your password: ";
}
else
{
cout << "Wrong password--try again: ";
}
cin >> password;
} while ( password != "xyzzzy" );
```

See how the do-while loop makes this more complicated, rather than less? The point is that the “body” is not the same—even though

This is the info i used to learn the loops.

Remix - July 30, 2014, 2:44 pm

A while loop is a head checked loop and a do while loop is a foot checked loop.

That means that a normal while loop checks the expression before executing the code.

A do while loop checks the expression after executing the code.

hackaquack - July 30, 2014, 2:45 pm

(Jul 30, 2014 02:44 PM)[iRemix] Wrote: [\[CITE\]](#) A while loop is a head checked loop and a do while loop is a foot checked loop.

That means that a normal while loop checks the expression before executing the code.

A do while loop checks the expression after executing the code.

And I'm going to assume if the expression results to false then the code doesn't execute right?

Remix - July 30, 2014, 3:00 pm

(Jul 30, 2014 02:45 PM)hackaquack Wrote: [\[CITE\]](#) And I'm going to assume if the expression results to false then the code doesn't execute right?

With a do while loop it will be executed at least once.

If the expression if false after the first execution it won't be executed again.

That's just because the expression will be checked after the code execution.

6577657534735 - July 30, 2014, 4:10 pm

PHP Code

```
<?php
do{
    //body
}while(x);
Does this:
//body
if(!x){break;}
//body
```

```
if(!x){break;}
//body
...etc
```

PHP Code

```
<?php
while(x){
```

```
}
```

Does this:

```
if(!x){break;}
//body
if(!x){break;}
//body
if(!x){break;}
...etc
```

newform12 - July 30, 2014, 4:26 pm

```
while(under_condition)
```

```
{
do stuff
}
```

```
//
do
{
do_this_atleast_once_before_while_loop_starts
} while (under_condition)
```

hackaquack - July 30, 2014, 4:43 pm

Would it be legal to do:

PHP Code

```
<?php
do {
```

```
    //body
```

```
} while(expression) {
```

```
    //body
```

```
};
```

So I can prevent what's being executed by the do statement from executing twice and execute the body of the while loop instead?

Thanks for the examples guys, they explained things a bit more.

newform12 - July 30, 2014, 4:47 pm

You basically just add one more expression before your while loop.

```
while(condition)
{
```

```
expression  
}
```

Irritable - July 30, 2014, 7:26 pm

(Jul 30, 2014 04:43 PM)hackaquack Wrote: [\[CITE\]](#) Would it be legal to do:

PHP Code

```
<?php  
do {  
  
    //body  
  
} while(expression) {  
  
    //body  
  
};
```

So I can prevent what's being executed by the do statement from executing twice and execute the body of the while loop instead?

Thanks for the examples guys, they explained things a bit more.
I'm not quite sure I follow your logic there.

A do-while statement is guaranteed to execute at a minimum of once
A while statement is not guaranteed to execute ever.

If a while statement is evaluated to be false from the start: It will never execute.
If a do-while statement is evaluated to be false from the start: It will execute exactly once.

0xDEAD10CC - July 30, 2014, 9:06 pm

(Jul 30, 2014 04:43 PM)hackaquack Wrote: [\[CITE\]](#) Would it be legal to do:

PHP Code

```
<?php  
do {  
  
    //body  
  
} while(expression) {  
  
    //body  
  
};
```

So I can prevent what's being executed by the do statement from executing twice and execute the body of the while loop instead?

Thanks for the examples guys, they explained things a bit more.
What kind of logic would that represent even if it was valid syntax? It isn't though, because the syntax for a do while loop includes a semi-colon after the while conditional...

Code do
{
}

```
while ();  
And even if you did that:  
Code do {
```

```
    //body
```

```
} while(expression); {
```

```
    //body
```

```
};  
It's not going to do as you'd expect.
```

You have the do while loop. Then the semi-colon separates that other scope from any loop semantics really... You end up with a manual scope denoted by the last { and }, and a redundant semicolon at the end of this scope... Thus, the do while loop would finish its thing, then the second body would execute ONCE.

DsntExist - July 30, 2014, 9:16 pm

Well if you want the code to run FIRST, and THEN check for the condition, go for the do while.

If you want the code to check the condition FIRST, and THEN execute. while loops are the way to go.

I was confused on the differences too when I first started coding, but the more you start building the more you'll start thinking to yourself "Ahh! this loop would be just right for this part of my code!".

Hopefully that helped. ? Happy coding!

hackaquack - July 31, 2014, 10:25 am

(Jul 30, 2014 09:06 PM)0xDEAD10CC Wrote: [\[CITE\]](#) What kind of logic would that represent even if it was valid syntax? It isn't though, because the syntax for a do while loop includes a semi-colon after the while conditional...

```
Code do  
{  
}  
while ();  
And even if you did that:  
Code do {
```

```
    //body
```

```
} while(expression); {
```

```
    //body
```

```
};  
It's not going to do as you'd expect.
```

You have the do while loop. Then the semi-colon separates that other scope from any loop semantics really... You end up with a manual scope denoted by the last { and }, and a redundant semicolon at the end of this scope... Thus, the do while loop would finish its thing, then the second

body would execute ONCE.

I know I was just asking. Ofc I'm not gonna do that. I was just wondering. :) Thanks!

Crypt0leucus - August 1, 2014, 2:06 pm

(Jul 30, 2014 02:24 PM)hackaquack Wrote: [\[CITE\]](#) I see no difference between a do while loop and a while loop.

A do while loop is basically a while loop with the loop body being declared before the expression.

What's the difference between

Code do {

statements

} while(expression);And

Code while(expression) {

statements

}

?

How can I make use of the do while loop?

Thanks,

- John

A while loop executes ZERO or more times and a do while loops is guaranteed to execute ONE or more times.

The while loop has a preloop test while the do while loop has a postloop test.

The do while can be used for error handling or forcing the user to input specific things.

Realistically you can rewrite all the control structures to do the same thing, so there isn't really much of a difference in code, just how you use it.

Masm4ever - August 1, 2014, 3:43 pm

(Jul 30, 2014 02:40 PM)Enju Wrote: [\[CITE\]](#) Do-while loops

These are the black swans of programming

So I'm using black swans all the time - probably my fault.

OTOH, they are slightly more efficient - one jump less.