Maximizing Efficiency in ServiceNow: .get() vs .next()

Ever wondered how to retrieve a single record from ServiceNow ?

Throughout my journey with ServiceNow I would approach this tasks very differently. Lets see what are ways to level up this code from **Beginner** to **Pro**

Query Single Record - Level Beginner

```
var gr = new GlideRecord('table_name');
gr.addQuery('field_name', 'value');
gr.query();

while (gr.next()) {
// operations on the record
}
```

If you have been in a ServiceNow for a while you immidiately notice that this is not the best way to query a single record because of a few things. Let’s explore them:

1. Using ‘if (gr.next())’ instead of ‘while (gr.next())’

if (gr.next()) is used when you're only interested in processing a single record. If a record is found that matches the query, the code inside the **if** block will be executed once, and then the program will move on.

This is useful when you're querying for a specific record and you know there will only be one match, or when you just want to check if any matching records exist.

while (gr.next()), on the other hand, is used when you want to process multiple records. The code inside the **while** block will be executed for each record that
matches the query, until there are no more matching records. This is useful when you're querying for all records that meet certain criteria and you want to do something with each one.

So, if you're only interested in a single record, **if (gr.next())** is better because it makes it clear that only one record will be processed, and it avoids unnecessary iterations.

If you used **while (gr.next())** and there were multiple matching records, it would process all of them, which could be inefficient and could lead to unexpected results if you're not expecting more than one match.

2. Using .setLimit(1)

The .setLimit(1) method limits the query to return only one record.

This can improve performance, because search is stopped when specified amount of records is found (in our example it's 1). Whenever you know number of records that you want to query, .setLimit is good choice, as it stops right after it finds desired number of records.

If you use **.setLimit(1)** with **if (gr.next())**, it will work as expected. The query will return at most one record, and if a record is found, the code inside the **if** block will be executed once.

If you use **.setLimit(1)** with **while (gr.next())**, the **while** loop will also only iterate once, because the query will only return at most one record due to the limit. So in this case, using **while** would be equivalent to using **if**.

However, using **while (gr.next())** with **.setLimit(1)** could be misleading to someone reading the code, because **while** is typically used when you want to process multiple records. If you know you only want one record, it's clearer to use **if (gr.next())**.

In summary, if you're only interested in a single record, it's better to use if (gr.next()) with .setLimit(1).

If you're interested in multiple records, you should use while (gr.next()), and only use .setLimit(1) if you want to limit the results to one record for some reason.

3. Using ‘.get()’

My personal favourite is using **.get()** to query a single record.

If you know unique identifier of record you are looking for like **sys_id** (but you can also use other like number of INC etc.) method can be faster and more efficient.

The **.get()** method retrieves one record based on the sys_id or a specified condition, and it stops querying once it finds a match.

But you must be **aware** of one thing !

Here's an example of using `.get()`:

```
// As you can see the syntax is pretty short, ficiency with fast implementation
var gr = new GlideRecord('table_name');
if (gr.get('sys_id')) { // or you can search 234')) {
    // operations on the record
}
```

Be aware that **‘if’** statement is crucial ! By wrapping the **.get()** method in an **if** statement, you're checking whether a record was actually returned before trying to access it.

If you don't do this and try to access fields on the GlideRecord object when **.get()** has returned **false**, you'll get a **null** reference error because there's no record to access.

While the **.get()** method can indeed be used to retrieve multiple records if used in conjunction with **.next()**, it's not the most efficient way to do so.

The **.get()** method is primarily designed to retrieve a single record based on a unique identifier (like sys_id) or a unique condition. If multiple records match the condition, **.get()** will only return the first one, and you would have to call **.next()** to access the others.

So, if you need to retrieve multiple records, it's recommended to use **.addQuery()/.query()** with a **while(gr.next())** loop. If you need to retrieve a single record based on a unique condition, use **.get()**.

Conclusions TL&DR:

1. Use **‘if’** when queryingwith single record, use **‘while’** when querying multiple records.

2. Use **‘.setLimit()’** whenever you know number of records that are to be returned. This way the search stops as soon as number of limited records is met. So for exmple when .setLimit(1) finds one record that matches the query it stops looking for more.

3. Use **‘.get()’** is the best way to retrieve a single record, as it looks throught database and stops at first record that it finds. It has also the shortest syntax.

If you like this post, **follow me** for more deep dives into **ServiceNow** and **Personal Development.**