"Apex Code Best Practices"

https://devloper.salesforce.com/page/Apex_Code_Best_Practices

https://developer.salesforce.com/index.php?title=Apex_Code_Best_Practices&oldid=26951

1: Bulkify your Code.
2: Avoid SOQL Queries or DML statements inside FOR Loops
3: Bulkify your Helper Methods
4: Using Collections, Streamlining Queries, and Efficient For Loops
5: Streamlining Multiple Triggers on the Same Object
6: Querying Large Data Sets
7: Use of the Limits Apex Methods to Avoid Hitting Governor Limits
8: Use @future Appropriately
9: Writing Test Methods to Verify Large Datasets
10: Avoid Hardcoding IDs

Since Apex is case insensitive you can write it however you'd like. However, to increase readability, follow Java capitalization standards and use two spaces instead of tabs for indentation.
Use Asynchronous Apex (@future annotation) for logic that does not need to be executed synchronously.
Asynchronous Apex should be "bulkified".
Apex code must provide proper exception handling.
Prevent SOQL and SOSL injection attacks by using static queries, binding variables, or the escape single quotes method.
When querying large data sets, use a SOQL "for" loop
Use SOSL over SOQL where possible - it's much faster.
Use Apex Limits Methods to avoid hitting governor exceptions.
No SOQL or SOSL queries inside loops
No DML statements inside loops
No Async (@future) methods inside loops
Do not use hardcoded IDs

===

Visualforce

Do not hardcode picklists in Visualforce pages; include them in the controller instead.
Javascript and CSS should be included as Static Resources allowing the browser to cache them.
Reference CSS at the top and JavaScript a the bottom of Visualforce pages as this provides for faster page loads.
Mark controller variables as "transient" if they are not needed between server calls. This will make your page load faster as it reduces the size of the View State.
Use <apex:repeat> to iterate over large collections.
Use the cache attribute with the <apex: page> component to take advantage of CDN caching when appropriate

==

Salesforce : cloud : Server : SaaS, PaaS, IaaS
Objects ==> Models===> Server
VFP ===> Controller ===> Server
Apex code ==> Saves, compile , error, Compiled Code, Exceution
Server ==> Resources ===> HD, RAM, Processor
Salesforce Server ==> Org (Millions)==> Busy

Multi-tenant environment ===>
Interaction ====>
We have to establish connection ===>
request (Client ----> Server)
A transaction begins
response (Server ---> Client)

Transaction is over
Governor Limits: SOQL ===> 100
SOSL ===> 20
DML ===> 150
Records SOQL ==> 50000
heap ==> 6 MB

=======>
Transaction
Synchronous: Immediate resources

Asynchornous : Heap : 12 MB
SOQL : 200

=======>

50000 records in a single transaction SOQL

1) [];
2) Database.query('')

Bank : SBI ===> millions accounts
At the end of the month, every account holder Bank
The statement has to be sent
===

Batch Class: Asynchronous
50 million records at a time can be processed.

Dataase.QueryLocator obj = Database.getQueryLocator('');

==

```
	global class FirstClass implements Database.Batchable{


		global Database.QueryLocator start(Database.BatchableContext BC){
			Dataase.QueryLocator obj = Database.getQueryLocator('SELECT
firstName, email from contact');
			return obj;
		}

		global void execute(Database.BatchableContext BC,List scope){
		}

		global void finish(Database.BatchableContext BC){

		}

	}
```
==

```
	FirstClass obj = new FirstClass();
	Database.executeBatch(ObjectOfBatchClass, [ScopeSize] );
	Database.executeBatch(obj, 500);
```

==

We can schedule a batch class

We have to define a scheduled class.
interface: Schedulable

```
global class scheduledBatchable implements Schedulable {
        global void execute(SchedulableContext sc) {
                FirstClass obj = new FirstClass();
                database.executebatch(b);
        }
}
scheduledBatchable objSchedule = new scheduledBatchable();
System.schedule('ScheduleJobName',  ''   ,objSchedule );
```

Cron expression

Apex Scheduler :
https://developer.salesforce.com/docs/atlas.en-us.222.0.apexcode.meta/apexcode/apex_scheduler.htm

Using Batch Apex :
https://developer.salesforce.com/docs/atlas.en-us.apexcode.meta/apexcode/apex_batch_interface.htm

==