The [ca] section is mandatory. Here we tell OpenSSL to use the options from the [CA_default] section.

```
[ ca ]
# `man ca`
default_ca = CA_default
```

##

The [CA_default] section contains a range of defaults. Make sure you declare the directory you chose earlier (/root/ca).

```
[ CA_default ]
# Directory and file locations.
dir               = /root/ca
certs             = $dir/certs
crl_dir           = $dir/crl
new_certs_dir     = $dir/newcerts
database          = $dir/index.txt
serial            = $dir/serial
RANDFILE          = $dir/private/.rand

# The root key and root certificate.
private_key       = $dir/private/ca.key.pem
certificate       = $dir/certs/ca.cert.pem

# For certificate revocation lists.
crlnumber         = $dir/crlnumber
crl               = $dir/crl/ca.crl.pem
crl_extensions    = crl_ext
default_crl_days  = 30

# SHA-1 is deprecated, so use SHA-2 instead.
default_md        = sha256

name_opt          = ca_default
cert_opt          = ca_default
default_days      = 375
preserve          = no
policy            = policy_strict
```

##

We’ll apply policy_strict for all root CA signatures, as the root CA is only being used to create intermediate CAs.

```
[ policy_strict ]
# The root CA should only sign intermediate certificates that match.
# See the POLICY FORMAT section of `man ca`.
countryName             = match
stateOrProvinceName     = match
organizationName        = match
organizationalUnitName  = optional
commonName              = supplied
```

```
emailAddress            = optional

######################################################################
###############
```
We'll apply policy_loose for all intermediate CA signatures, as the intermediate CA is signing server and client certificates that may come from a variety of third-parties.
```
[ policy_loose ]
# Allow the intermediate CA to sign a more diverse range of certificates.
# See the POLICY FORMAT section of the `ca` man page.
countryName             = optional
stateOrProvinceName     = optional
localityName            = optional
organizationName        = optional
organizationalUnitName  = optional
commonName              = supplied
emailAddress            = optional

######################################################################
####################
```
Options from the [req] section are applied when creating certificates or certificate signing requests.
```
[ req ]
# Options for the `req` tool (`man req`).
default_bits        = 2048
distinguished_name  = req_distinguished_name
string_mask         = utf8only

# SHA-1 is deprecated, so use SHA-2 instead.
default_md          = sha256

# Extension to add when the -x509 option is used.
x509_extensions     = v3_ca

######################################################################
#########################
```
The [req_distinguished_name] section declares the information normally required in a certificate signing request. You can optionally specify some defaults.
```
[ req_distinguished_name ]
# See <https://en.wikipedia.org/wiki/Certificate_signing_request>.
countryName                     = Country Name (2 letter code)
stateOrProvinceName             = State or Province Name
localityName                    = Locality Name
0.organizationName              = Organization Name
organizationalUnitName          = Organizational Unit Name
commonName                      = Common Name
emailAddress                    = Email Address

# Optionally, specify some defaults.
```

```
                                    about_opensslCnf
countryName_default             = GB
stateOrProvinceName_default     = England
localityName_default            =
0.organizationName_default      = Alice Ltd
#organizationalUnitName_default =
#emailAddress_default           =

########################################################################
########################
```

The next few sections are extensions that can be applied when signing certificates. For example, passing the -extensions v3_ca command-line argument will apply the options set in [v3_ca].

We'll apply the v3_ca extension when we create the root certificate.

```
[ v3_ca ]
# Extensions for a typical CA (`man x509v3_config`).
subjectKeyIdentifier = hash
authorityKeyIdentifier = keyid:always,issuer
basicConstraints = critical, CA:true
keyUsage = critical, digitalSignature, cRLSign, keyCertSign

########################################################################
################################
```

We'll apply the v3_ca_intermediate extension when we create the intermediate certificate. pathlen:0 ensures that there can be no further certificate authorities below the intermediate CA.

```
[ v3_intermediate_ca ]
# Extensions for a typical intermediate CA (`man x509v3_config`).
subjectKeyIdentifier = hash
authorityKeyIdentifier = keyid:always,issuer
basicConstraints = critical, CA:true, pathlen:0
keyUsage = critical, digitalSignature, cRLSign, keyCertSign

########################################################################
################################
```

We'll apply the usr_cert extension when signing client certificates, such as those used for remote user authentication.

```
[ usr_cert ]
# Extensions for client certificates (`man x509v3_config`).
basicConstraints = CA:FALSE
nsCertType = client, email
nsComment = "OpenSSL Generated Client Certificate"
subjectKeyIdentifier = hash
authorityKeyIdentifier = keyid,issuer
keyUsage = critical, nonRepudiation, digitalSignature, keyEncipherment
extendedKeyUsage = clientAuth, emailProtection

########################################################################
####################################
```

We'll apply the server_cert extension when signing server certificates, such as those used for web servers.

```
[ server_cert ]
# Extensions for server certificates (`man x509v3_config`).
basicConstraints = CA:FALSE
nsCertType = server
nsComment = "OpenSSL Generated Server Certificate"
subjectKeyIdentifier = hash
authorityKeyIdentifier = keyid,issuer:always
keyUsage = critical, digitalSignature, keyEncipherment
extendedKeyUsage = serverAuth

####################################################################
########################################
```

The crl_ext extension is automatically applied when creating certificate revocation lists.

```
[ crl_ext ]
# Extension for CRLs (`man x509v3_config`).
authorityKeyIdentifier=keyid:always

####################################################################
########################################
```

We'll apply the ocsp extension when signing the Online Certificate Status Protocol (OCSP) certificate.

```
[ ocsp ]
# Extension for OCSP signing certificates (`man ocsp`).
basicConstraints = CA:FALSE
subjectKeyIdentifier = hash
authorityKeyIdentifier = keyid,issuer
keyUsage = critical, digitalSignature
extendedKeyUsage = critical, OCSPSigning

####################################################################
########################################
```