HTTPS

To create an application that uses the HTTPS protocol, set the `httpsOptions` property in the options object passed to the `create()` method of the `NestFactory` class:

```
const httpsOptions = {
  key: fs.readFileSync('./secrets/private-key.pem'),
  cert: fs.readFileSync('./secrets/public-certificate.pem'),
};
const app = await NestFactory.create(AppModule, {
  httpsOptions,
});
await app.listen(3000);
```

If you use the `FastifyAdapter`, create the application as follows:

```
const app = await NestFactory.create<NestFastifyApplication>(
  AppModule,
  new FastifyAdapter({ https: httpsOptions }),
);
```

Multiple simultaneous servers

The following recipe shows how to instantiate a Nest application that listens on multiple ports (for example, on a non-HTTPS port and an HTTPS port) simultaneously.

```
const httpsOptions = {
  key: fs.readFileSync('./secrets/private-key.pem'),
  cert: fs.readFileSync('./secrets/public-certificate.pem'),
};

const server = express();
const app = await NestFactory.create(
  AppModule,
  new ExpressAdapter(server),
);
await app.init();

const httpServer = http.createServer(server).listen(3000);
const httpsServer = https.createServer(httpsOptions, server).listen(443);
```

Because we called `http.createServer` / `https.createServer` ourselves, NestJS doesn't close them when calling `app.close` / on termination signal. We need to do this ourselves:

```
@Injectable()
export class ShutdownObserver implements OnApplicationShutdown {
  private httpServers: http.Server[] = [];

  public addHttpServer(server: http.Server): void {
    this.httpServers.push(server);
  }

  public async onApplicationShutdown(): Promise<void> {
    await Promise.all(
      this.httpServers.map((server) =>
        new Promise((resolve, reject) => {
          server.close((error) => {
            if (error) {
              reject(error);
            } else {
              resolve(null);
            }
          });
        })
      ),
    );
  }
}

const shutdownObserver = app.get(ShutdownObserver);
shutdownObserver.addHttpServer(httpServer);
shutdownObserver.addHttpServer(httpsServer);
```

> info **Hint** The `ExpressAdapter` is imported from the `@nestjs/platform-express` package. The `http` and `https` packages are native Node.js packages.

> **Warning** This recipe does not work with GraphQL Subscriptions.