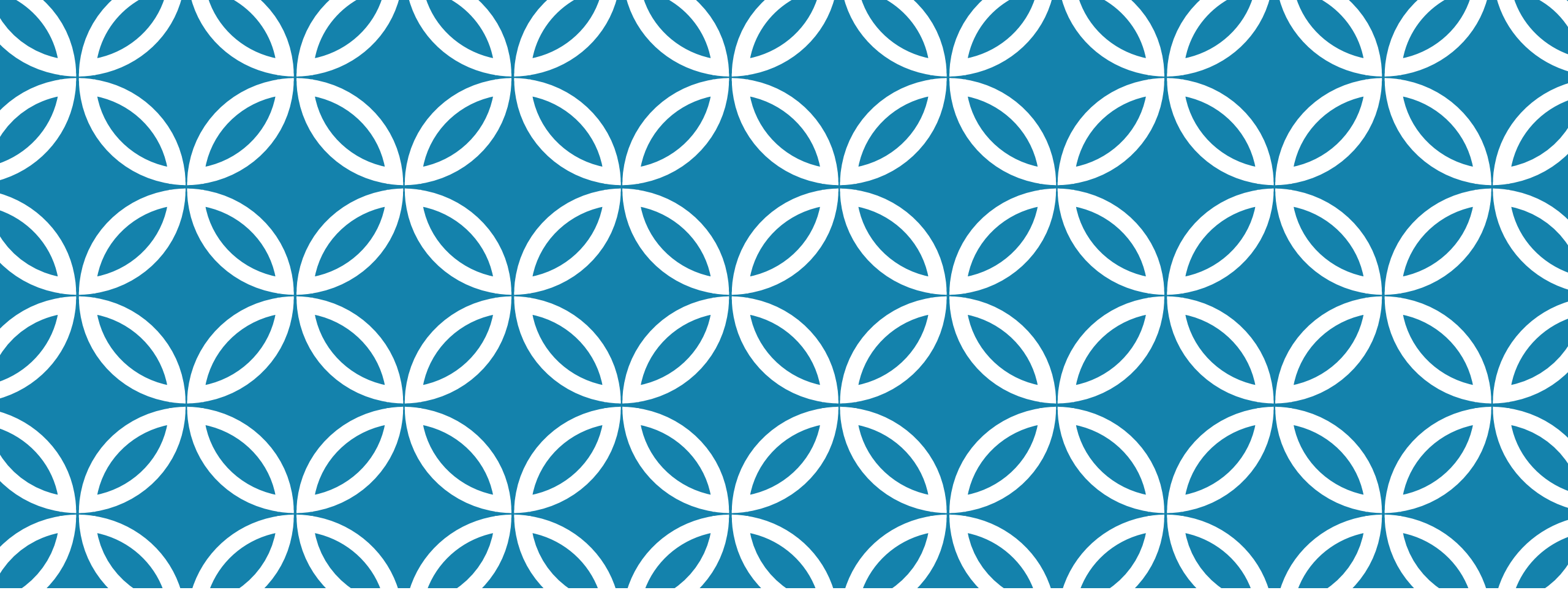
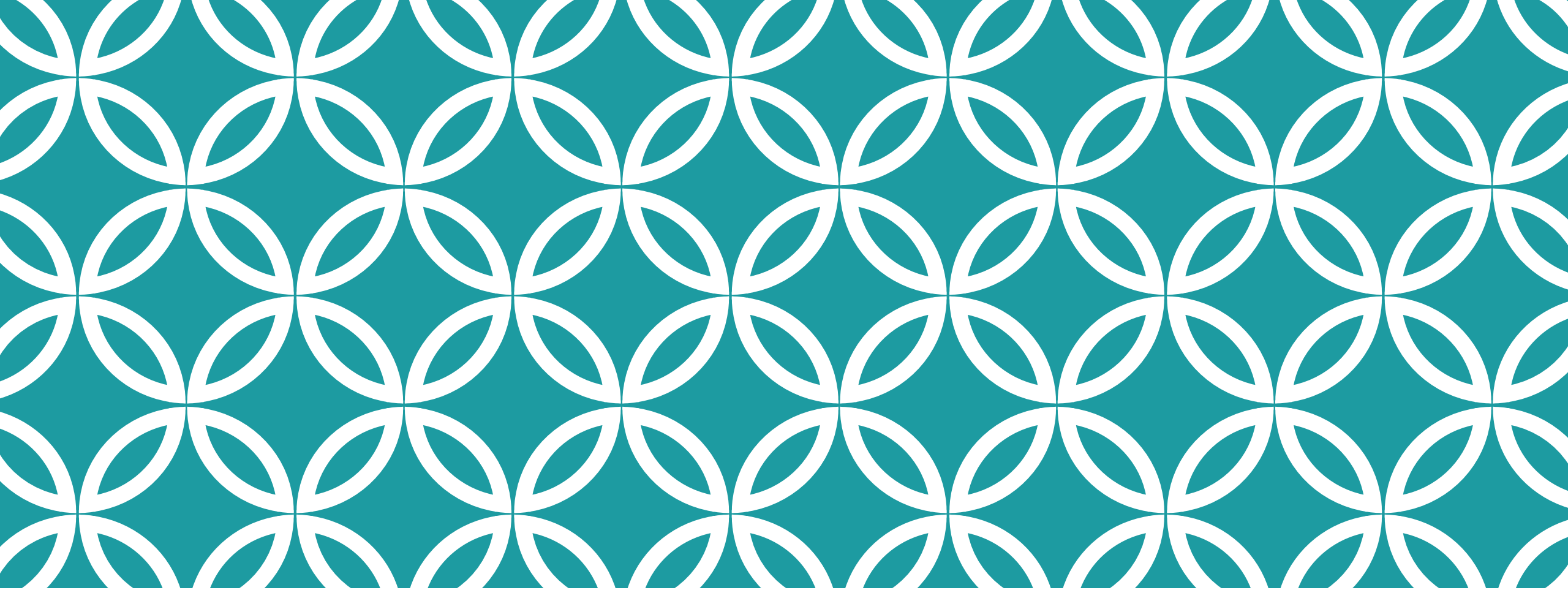


SIKKER KODE
USIKKER KODE

NAMESPACES
ASSEMBLIES

Grundlæggende programmering
Lektion 13



SIKKER KODE

Hvordan kan ens kode kaldes sikker

SIKKER KODE

Evidensbaseret sikkerhedspolitik og kode adgang sikkerhed sørger for meget kraftige, eksplicitte mekanismer til at implementere sikkerhed.

Den meste programkode kan blot bruge infrastrukturen fra .NET Framework.

I nogle tilfælde er yderligere applikationsspecifik sikkerhed, der kræves bygget enten ved at udvide sikkerheds-systemet eller ved hjælp af nye ad hoc-metoder, dog nødvendig.

Ved brug af .NET Framework tilladelser og andre håndhævelser i din kode, skal du opføre barrierer for at forhindre skadelig kode i at opnå oplysninger, som du ikke ønsker, at den skal have eller udfører andre uønskede handlinger.

SIKKER KODE

Desuden skal du finde en balance mellem sikkerhed og brugervenlighed i alle de forventede scenarier med betroet kode.

De næste slides vil gennemgå forskellige måder kode kan være konstrueret til at arbejde med sikkerhedssystemet på.

SIKKER KODE

SECURITY NEUTRAL KODE

Security-neutral kode gør ikke noget eksplicit med sikkerheds-systemet.

Den kører med de tilladelser den måtte modtage.

Selvom applikationer, der undlader at fange sikkerheds-undtagelser forbundet med beskyttede operationer (såsom at bruge filer, netværk, og så videre) kan resultere i en ikke-afviklet undtagelse, vil sikkerheds-neutral kode stadig udnytte .NET Framework sikkerhedsteknologierne.

SIKKER KODE

KODE DER IKKE ER GENBRUGELIG

Hvis din kode er en del af et program, der ikke bliver kaldt af anden kode, er sikkerhed enkel og speciel kodning er måske ikke være påkrævet.

Husk dog at skadelig kode kan kalde din kode.

Mens kode adgang sikkerhed kan stoppe skadelig kode i at få adgang til ressourcer, kan en sådan kode stadig læse værdier af dine fields eller egenskaber, der kan indeholde følsomme oplysninger.

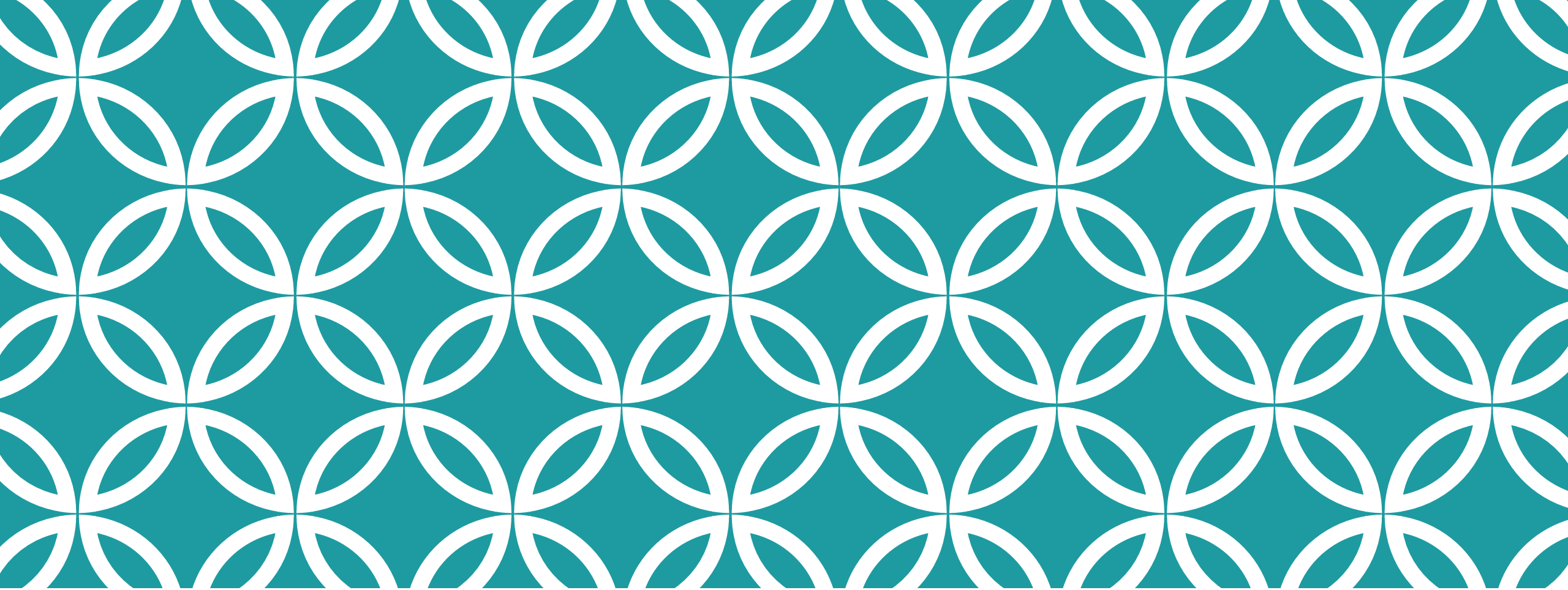
Desuden, hvis din kode accepterer brugerinput fra internettet eller andre upålidelige kilder, skal du være forsigtig med ondsindet input.

SIKKER KODE

BIBLIOTEKS KODE, DER UDSÆTTER BESKYTTEDE RESSOURCER

Dette er den mest kraftfulde og dermed potentielt farlige (hvis det gøres forkert) tilgang til sikker kodning: Dit bibliotek fungerer som en grænseflade til anden kode for at få adgang til visse ressourcer, der ellers ikke tilgængelige, ligesom klasserne af .NET Framework håndhæver tilladelser for de ressourcer, de bruger.

Uanset hvor du blotlægger en ressource, skal din kode først kræve hensigtsmæssig tilladelse til ressourcen (dvs., skal det foretage en sikkerhedskontrol) og derefter typisk udnytte sine rettigheder til at udføre selve operationen.



USIKKER KODE

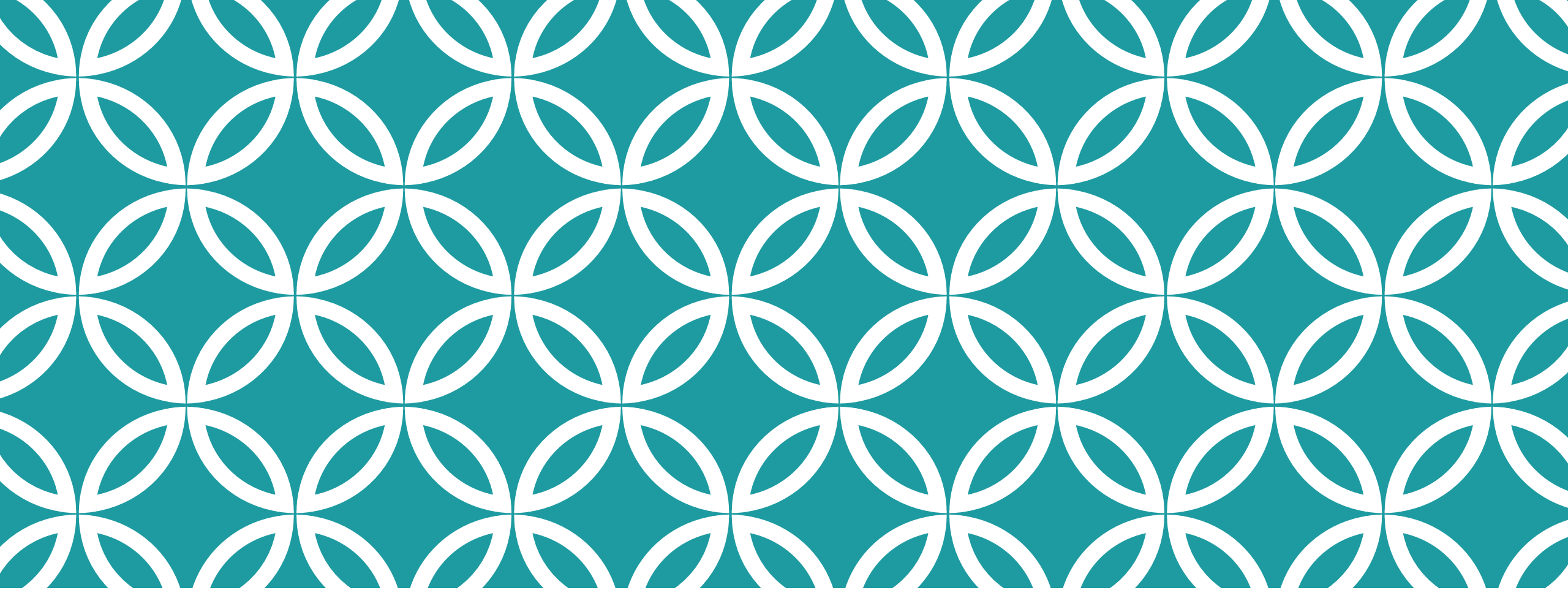
Hvordan kan ens kode kaldes usikker

USIKKER KODE

Hvis du vil ud over den sikkerhed som C# i sig selv tilbyder kan man gøre det på tre måder:

- Platform Invoke (P/Invoke)
  - Går gennem APIer er blotlægges af ikke-administrerede DLLer.
- Usikker kode
  - Tillader adgang til memory pointers og adresser.
- Windows Runtime (WinRT) API
  - Kun tilgængelig i Windows 8 eller nyere.
  - Lader en se mere af operativsystemets funktioner.

Ikke noget jeg vil anbefale i de fleste tilfælde, men det kan lade sig gøre.

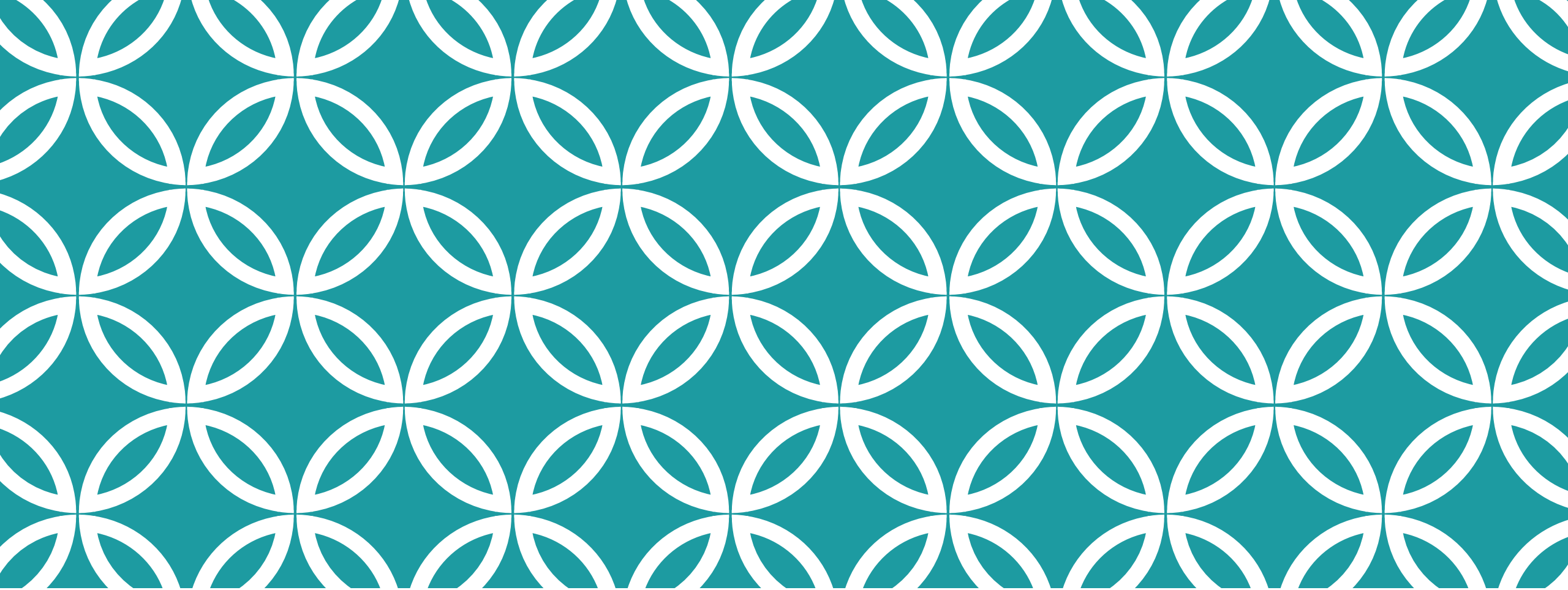


NAMESPACES

Mere om hvad namespaces er og hvordan de kan bruges

NAMESPACES

Et navneområde (**namespace**) er designet til at give en måde at holde et sæt navne adskilt fra en anden. Klassenavne, der er angivet i et navneområde, er ikke i konflikt med de samme klassenavne, der er angivet i et andet.



ASSEMBLIES

Samlet kode

ASSEMBLIES

Et assembly er en fil, der er automatisk genereres af compileren efter en vellykket samling af hver NET applikation.

Det kan enten være et Dynamic Link Library eller en eksekverbar fil.

Det genereres kun en gang for et program og ved hver efterfølgende kompilering bliver assembly opdateret.

Hele processen vil køre i baggrunden af ens applikation.

Et assembly indeholder Intermediate Language (IL) kode, som svarer til Javas byte kode.

I .NET sproget består det af metadata. Metadata enumerates ses i hver "type" inde i assembliet eller binary.

ASSEMBLIES

Ud over metadata har assemblies også en særlig ”fil” kaldet Manifest. Den indeholder oplysninger om den aktuelle version af assembliet og andre relaterede oplysninger.

I .NET, er der to slags assemblies: Single fil og Multi-fil.

En enkelt fil assembly indeholder alle de nødvendige oplysninger (IL, Metadata, og Manifest) i en enkelt pakke.

De fleste assemblies i .NET består af enkelt fil assemblies.

Multi fil assemblies er sammensat af en lang række NET binære filer eller moduler og genereres til større applikationer. Et assembly vil indeholde et manifest og andre vil have IL og Metadata instruktioner.

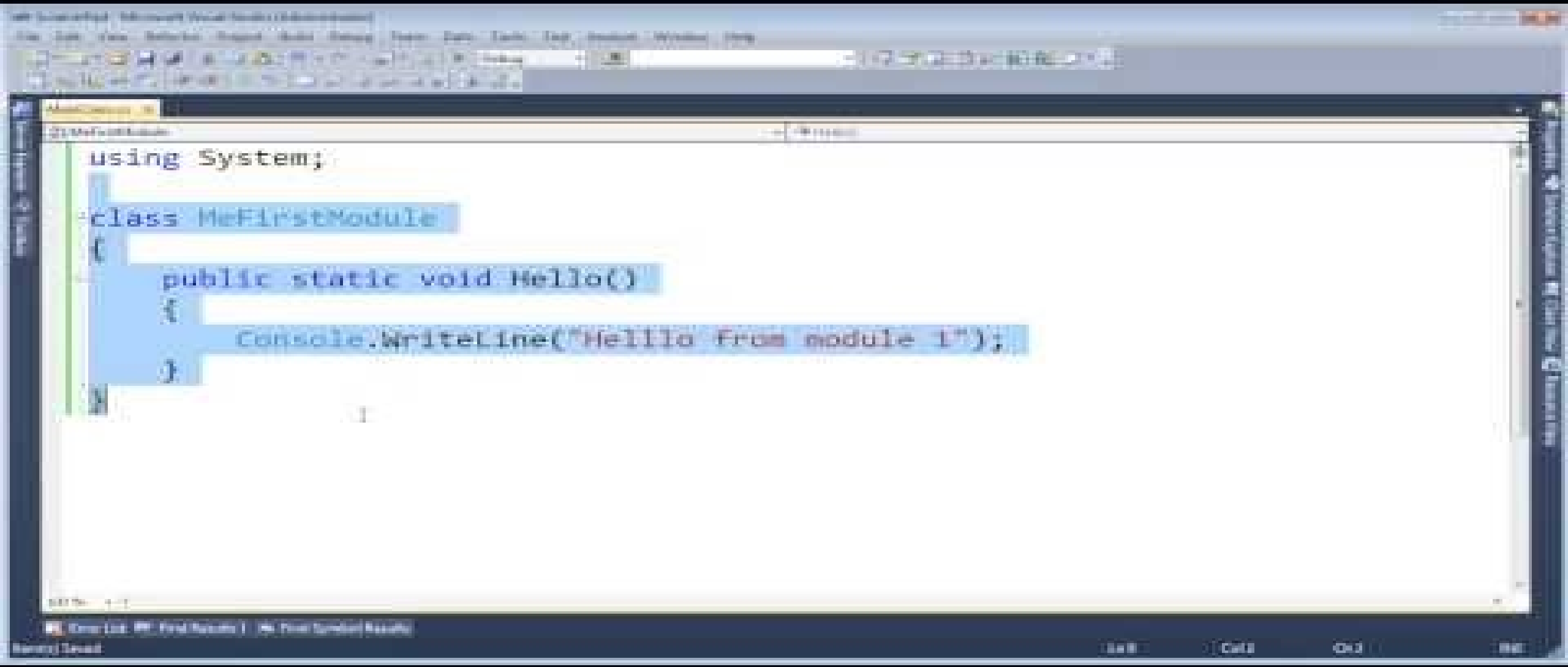
ASSEMBLIES

De fleste C# assemblies er som sagt EXE filer, men der er også library assemblies, der er DLL-filer.

De omtales ofte som klasse biblioteker, fordi de indeholder klasser, som du kan bruge i dine programmer.

Namespaces spiller en stor rolle i, hvordan man bruger dem.

En nem måde at lave en library collection er at lave et Class Library projekt i Visual Studio ved at vælge "Class Library" som projekt type, når du opretter et nyt projekt.


using System;
class MeFirstModule
{
public static void Hello()
{
Console.WriteLine("Hellio From module 1");
}
}

LEKTIE

Kig på dette til næste gang

LEKTIE

Læs

- https://www.tutorialspoint.com/csharp/csharp_unsafe_codes.htm
- https://www.tutorialspoint.com/csharp/csharp_namespaces.htm
- https://msdn.microsoft.com/da-dk/library/mt632255.aspx

Opgave

- Arbejd på jeres eksamensopgave

KILDER

Materiale benyttet i denne lektion
Noget af det er udover pensum-listen!

KILDER

Sikker kode

- https://msdn.microsoft.com/en-us/library/d55zzx87(v=vs.90).aspx
- https://msdn.microsoft.com/en-us/library/8a3x2b7f(v=vs.90).aspx

Assemblies

- http://www.codeguru.com/columns/csharp_learning/article.php/c5845/C-FAQ-15--What-is-an-Assembly.htm
- http://broadcast.oreilly.com/2010/07/understanding-c-namespaces-and.html
- https://youtu.be/ky3_3ubDxII