Using Shared Access Signatures

Thomas Claudius Huber
SOFTWARE DEVELOPER

@thomasclaudiush www.thomasclaudiushuber.com

Module Outline

ASP.NET Core web app

- Use the Blob URL

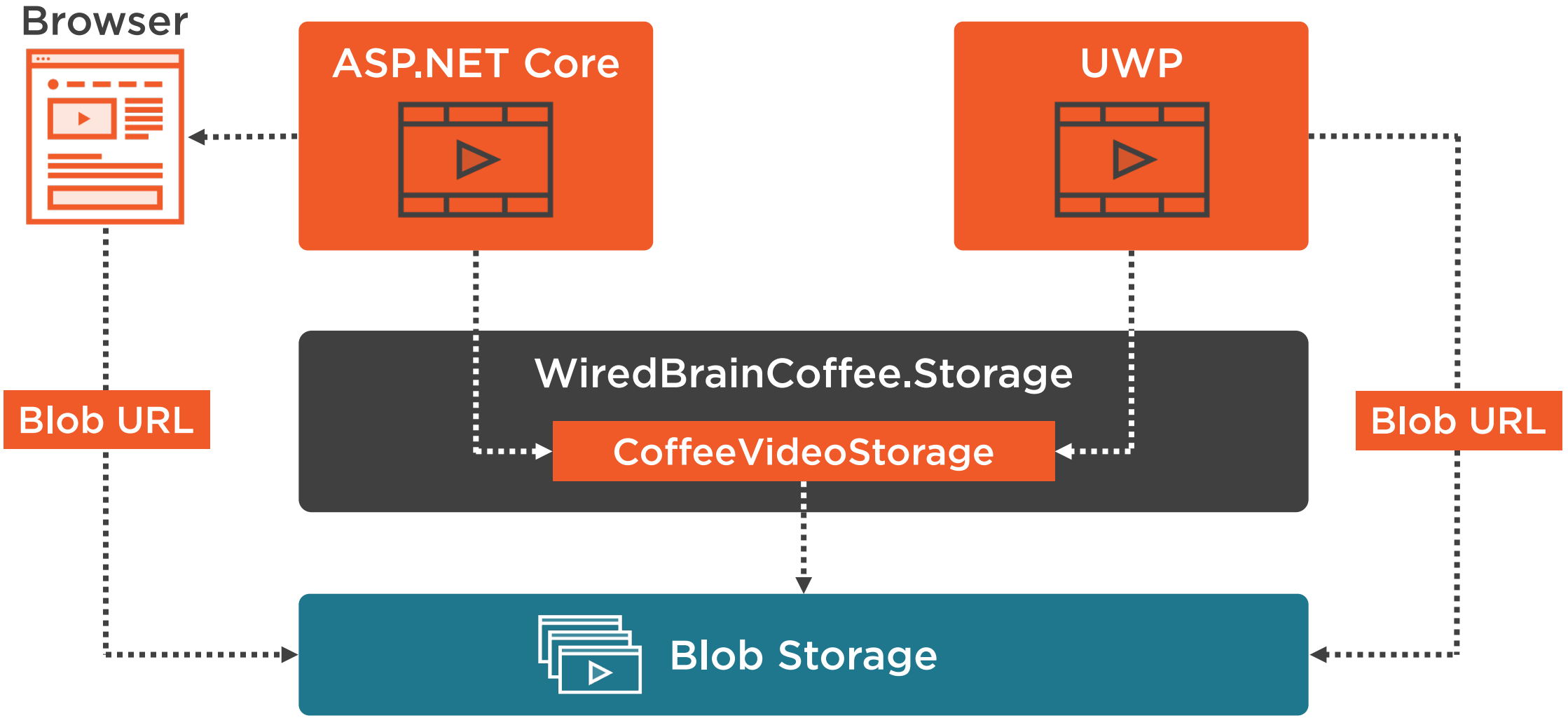
Set the Container's public access level to private

Access Blobs with a Shared Access Signature (SAS)

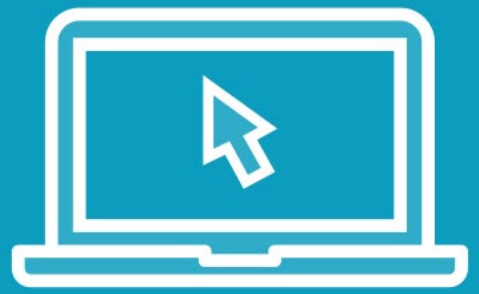
- Create a SAS in the Azure Portal
- Create a SAS in .NET

Know the types of Shared Access Signatures

Use the Blob URL in ASP.NET Core



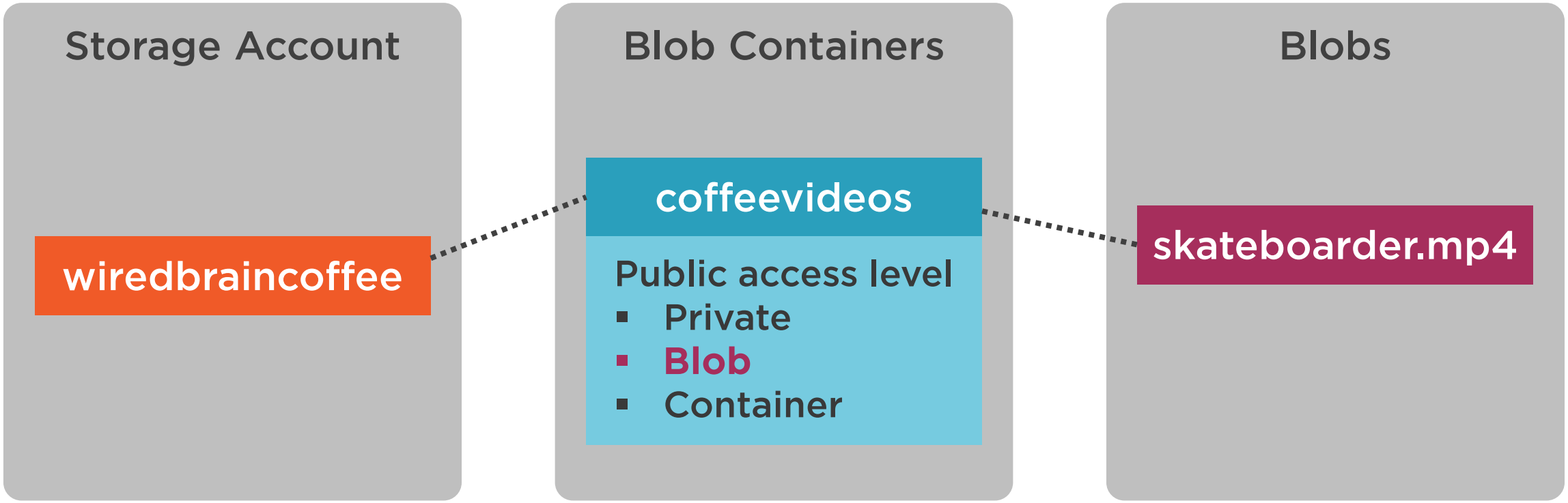
Demo



The ASP.NET Core web application

Use the Blob URL

Set the Public Access Level to Private

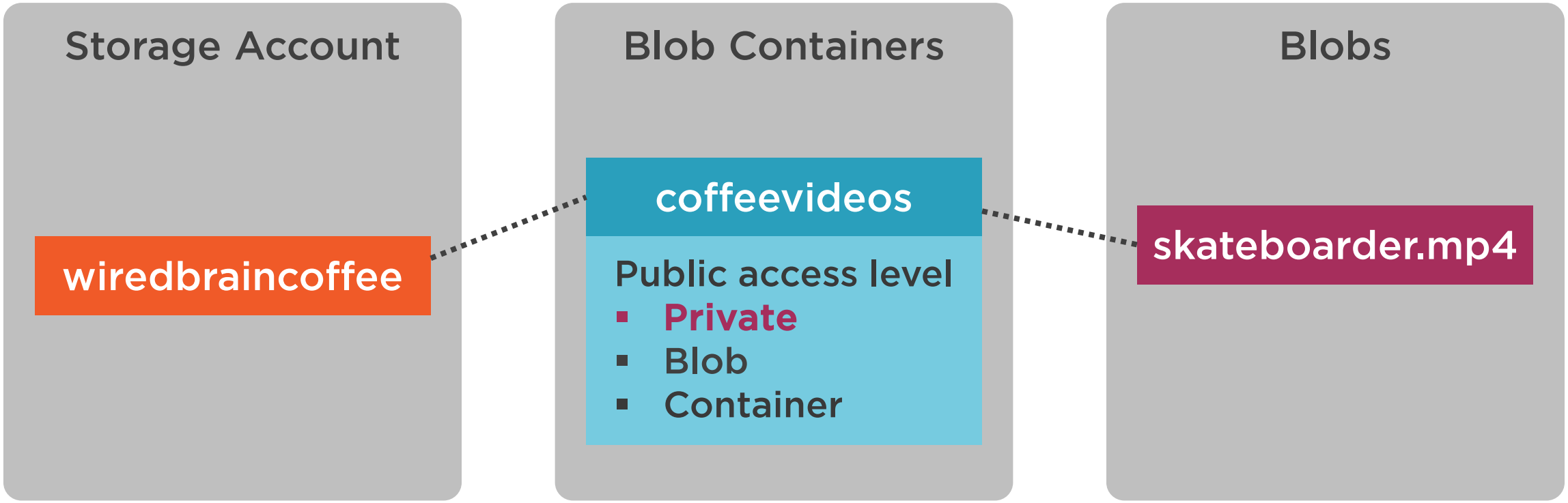


https://wiredbraincoffee.blob.core.windows.net/coffeevideos/skateboarder.mp4

“We don’t want that anyone can use the Blob URLs in their static website.”

Sara – software developer at Wired Brain Coffee

Set the Public Access Level to Private



https://wiredbraincoffee.blob.core.windows.net/coffeevideos/skateboarder.mp4

Set the Public Access Level to Private

```
await cloudBlobContainer.CreateIfNotExistsAsync(
    BlobContainerPublicAccessType.Blob, null, null);
```

Set the Public Access Level to Private

```
await cloudBlobContainer.CreateIfNotExistsAsync(
    BlobContainerPublicAccessType.Off, null, null);
```

Set the Public Access Level to Private

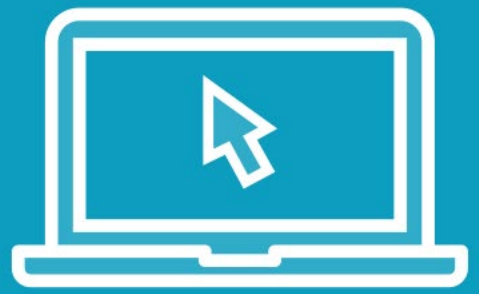
```
await cloudBlobContainer.CreateIfNotExistsAsync();


var blobContainerPermissions = new BlobContainerPermissions
{
  PublicAccess = BlobContainerPublicAccessType.Off
};
await
cloudBlobContainer.SetPermissionsAsync(blobContainerPermiss
ions);
```

Set the Public Access Level to Private

```
await cloudBlobContainer.CreateIfNotExistsAsync();
```

Set the public access level to private in the Azure Portal

Demo



Adjust the call of the CreateIfNotExistsAsync method

Set the public access level on the Container to private

Access Blobs with a Shared Access Signature

Anonymous Access

Account Key

Role-based
Access Control (RBAC)

Shared Access Signatures
(SAS)

Access Blobs with a Shared Access Signature

A Shared Access Signature is a signed URL to access a resource

https://wiredbraincoffee.blob.core.windows.net/coffeevideos/skateboarder.mp4
?sp=rcwd&st=2018-09-28T18:15:28Z&se=2018-09-29T02:15:28Z&spr=https
&sv=2017-11-09&sig=w2pECW9KbgQhGs6eMrj%2BG7ggSbxndcE8jkotk669KYo
%3D&sr=b

Access Blobs with a Shared Access Signature

https://wiredbraincoffee.blob.core.windows.net/coffeevideos/skateboarder.mp4
?sp=rcwd&st=2018-09-28T18:15:28Z&se=2018-09-29T02:15:28Z&spr=https
&sv=2017-11-09&sig=w2pECW9KbgQhGs6eMrj%2BG7ggSbxndcE8jkotk669KYo
%3D&sr=b

- **sp** — Permissions
- **st** — Start time
- **se** — Expiry time
- **spr** — Protocol
- **sv** — Storage service version
- **sig** — Signature created with SHA256 algorithm
- **sr** — Storage resource

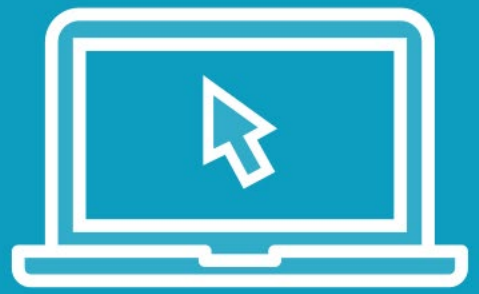
Access Blobs with a Shared Access Signature

Shared Access Signatures are created client-side

Blob Service does NOT know the number of Shared Access Signatures

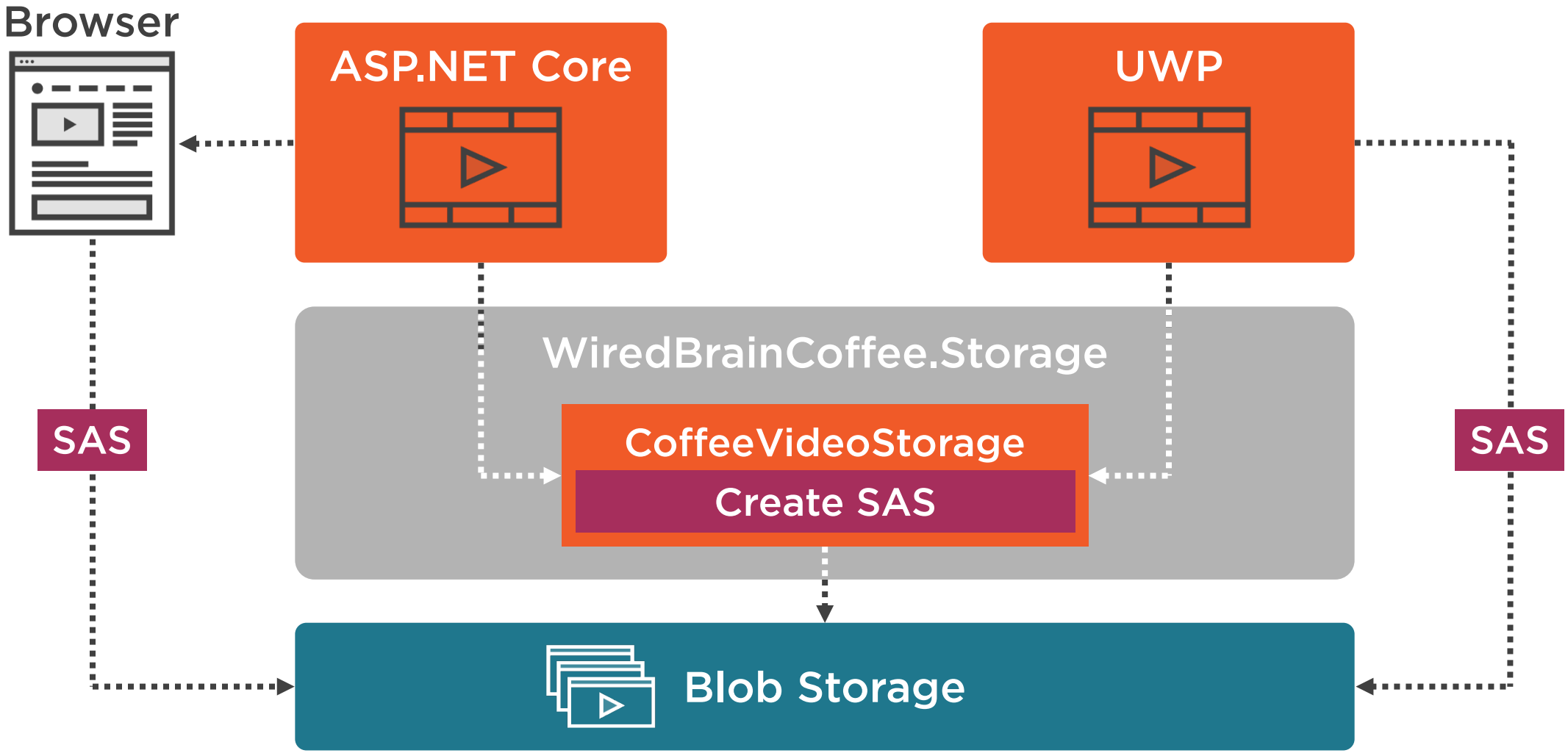
The expiry time is important

Demo

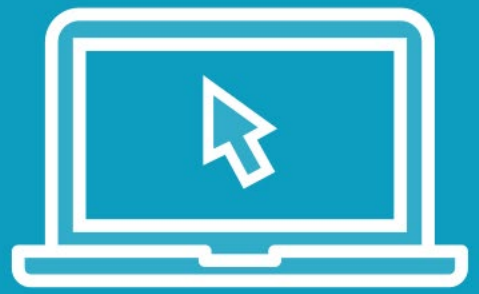


Create a Shared Access Signature for a Blob in the Azure Portal

Create a Shared Access Signature in .NET



Demo



Create a Shared Access Signature in the CoffeeVideoStorage class

Use the SAS

- In the ASP.NET Core app
- In the UWP app

Know the Types of Shared Access Signatures

Service

Shared Access Signatures

Account

Shared Access Signatures

Know the Types of Shared Access Signatures

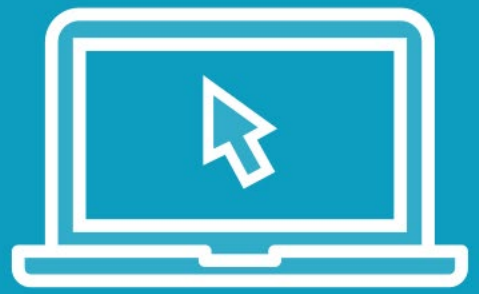
Blob Service SAS

```
cloudBlockBlob.GetSharedAccessSignature(sharedAccessBlobPolicy);

cloudBlobContainer.GetSharedAccessSignature(sharedAccessBlobPolicy);
```

Account SAS

```
cloudStorageAccount.GetSharedAccessSignature(sharedAccessAccountPolicy);
```

Demo



Create an Account SAS in the Azure Portal

Summary

Access Blobs with a Shared Access Signature

- Blob URL + SAS token

Create a Shared Access Signature

- In the Azure Portal
- In .NET

Types of Shared Access Signatures

- Service SAS
- Account SAS

URL to read a Blob is valid for 24 hours