



PODSTAWOWE INTERFEJSY TRWAŁOŚCI:

- IPersist
- IPersistStream
- IPersistStreamInit
- IPersistStorage
- IPersistFile
- IPersistPropertyBag

PODSTAWOWY INTERFEJS IPERSIST

```
interface IPersist : IUnknown
{
    HRESULT GetClassID([out] CLSID *pClsId);
};
```

OBSŁUGA STRUMIENI

```
interface IPersistStreamInit : IPersist
{
    HRESULT IsDirty();
    HRESULT Load([in, unique] IStream *pStr);
    HRESULT Save([in, unique] IStream *pStr,
                [in] BOOL fClearDirty);
    HRESULT GetSizeMax([out]
                       ULARGE_INTEGER *pSize);
    HRESULT InitNew();
};
```

```
interface ISequentialStream : IUnknown
{
    HRESULT Read([out, size_is(cb),
                length_is(*pRead)] void *p,
                [in] ULONG cb,
                [out] ULONG *pRead);
    HRESULT Write(
                [out, size_is(cb)] void *p,
                [in] ULONG cb,
                [out] ULONG *pWritten);
};
```

ZAPIS TRWAŁEGO OBIEKTU

```
// pobieramy clsid klasy
CLSID clsid;
hr = pPersistent->GetClassID(&clsid);

// zapisujemy clsid (standardowa funkcja)
hr = WriteClassStm(pStream, clsid)

// zapisujemy dane obiektu
hr = pPersistent->Save(pStream, TRUE);
```

ODCZYT TRWAŁEGO OBIEKTU

```
// pobieramy CLSID
CLSID clsid;
hr = ReadClassStm(pStream, &clsid);

// tworzymy nowe wystąpienie obiektu
IPersistStream *pPersistent;
hr = CoCreateInstance(clsid, NULL,
                    CLSCTX_INPROC_SERVER, IID_IPersistStream,
                    (void**)pPersistent);

// wciągamy dane obiektu
pPersistent->Load(pStream);
```

SPRYTNE FUNKCJE STANDARDOWE

```
hr = OleSaveToStream(pPersistent, pStream);

hr = OleLoadFromStream(pPersistent, pStream);
```

PRACA Z ZESTAWAMI WŁAŚCIWOŚCI

```
interface IPersistPropertyBag : IPersist
{
    HRESULT IsDirty();
    HRESULT Load([in] IPropertyBag *pPropBag,
                [in] IErrorLog *pErrorLog);
    HRESULT Save([in] IPropertyBag *pPropBag,
                [in] BOOL fClearDirty,
                [in] BOOL bSaveAllProps);
    HRESULT GetSizeMax([out]
                       ULARGE_INTEGER *pSize);
    HRESULT InitNew();
};
```

```
interface IPropertyBag : IUnknown
{
    HRESULT Read([in] LPCOLESTR pPropName,
                [in, out] VARIANT *pVar,
                [in] IErrorLog *pErrorLog);

    HRESULT Write([in] LPCOLESTR pPropName,
                [in] VARIANT *pVar);
};
```