

Sprachausgabe und Spracherkennung

Thomas Reinwart

Bei Sprachinteraktion unterscheidet man die **Spracherkennung** (SR: *Speech recognition*) und die **Sprachsynthese** (TTS: *Text to speech*). Die akustische Interaktion kann genutzt werden, um dem Rechner Befehle zu erteilen, die er auszuführen hat. Aber auch das Vorlesen von Texten ist möglich. Diese Möglichkeiten kommen auch Blinden oder körperbehinderten Menschen zugute.

Microsoft Speech Application Programming Interface (SAPI)

Bei der SAPI handelt es sich um eine Sprachinteraktionsmöglichkeit auf Windows PCs. SAPI gibt seit Windows 95/NT 3.51. SAPI wird für Windows entweder als eigene *Speech SDK* angeboten oder wird bereits mit Windows selbst mitgeliefert. Sie kann frei verwendet werden.

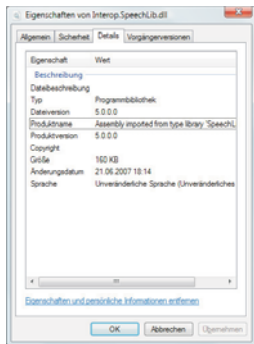
Microsoft Produkte die SAPI unterstützen sind *Microsoft Office Agent* und der *Speech Server*.

SAPI Versionen

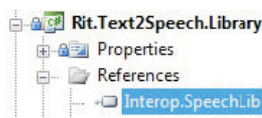
Die SAPI Versionen 1–4 sind im Wesentlichen gleich vom Aufbau, jede neue Version bietet neuen Funktionen. SAPI 5 hat ein neues Interface bekommen. Die aktuelle Version ist 5.3.

- SAPI 1.0: 1995: erstes Release für Windows 95 und Windows NT 3.51
- SAPI 2.0: 1996: Erweiterungen
- SAPI 3.0: 1997: Unterstützung für den Diktat Modus
- SAPI 4.0: 1998: COM API, ActiveX Controls für VB und C++ Wrapper Klassen
- SAPI 5.0: 2000: Redesign mit Runtime sapi.dll als Engine; reine COM API; ab Windows 98 und NT 4.0
- SAPI 5.1: 2001: Automation-compliant interfaces in der API; wurde mit Windows XP ausgeliefert
- SAPI 5.2: 2004: API für den Microsoft Speech Server
- SAPI 5.3: Aktuelle Version in Windows Vista

SAPI 5.0



Das nebenstehende Beispiel zeigt die Verwendung der SAPI 5.0 in einem C#.net-Projekt. Dazu wird die COM *speechlib.dll* der SAPI SDK als Referenz eingebunden.

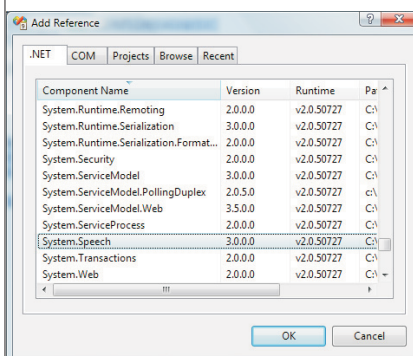


Speech-Synthesizer Kodebeispiel SAPI 5.0 (C# Windows Forms)

```
using SpeechLib;
public static void SpeakText(string text)
{
    SpeechLib.SpVoice voice = new SpVoice();
    // Achtung falls voice gewählt wird immer "Name=" davorsetzen
    voice.Speak(text, SpeechVoiceSpeakFlags.SVSFDefault);
}
public static List<string> GetVoices()
{
    SpeechLib.SpVoice voice = new SpVoice();
    SpeechLib.ISpeechObjectTokens listOfVoices = voice.GetVoices("", "");
    List<string> voices = new List<string>();
    for (int i = 0; i < listOfVoices.Count; i++)
    {
        voices.Add(listOfVoices.Item(i).GetAttribute("Name"));
    }
    return voices;
}
```

```
public static string
SaveToFile(string text, int volume, string voicename, string language)
{
    SpeechLib.SpVoice voice = new SpVoice();
    SpeechLib.SpFileStream str = new SpFileStream();
    System.Windows.Forms.SaveFileDialog dialog = new SaveFileDialog();
    dialog.Filter = "Wave Files (*.wav)|*.wav";
    if (dialog.ShowDialog() == DialogResult.OK)
    {
        try
        {
            str.Open(dialog.FileName,
                SpeechLib.SpeechStreamFileMode.SSFMCreateForWrite, false);
            language.Item(0);
            voice.Volume = volume;
            voice.Speak(text, SpeechLib.SpeechVoiceSpeakFlags.SVSFDefault);
        }
        finally
        {
            str.Close();
        }
    }
    return dialog.FileName;
}
```

SAPI 5.3



Das Beispiel auf der folgenden Seite zeigt die Verwendung der aktuellen SAPI 5.3 in einem C#.net Projekt. Der Unterschied zur vorherigen Version ist, dass mit der Installation von .net-Framework 3.0 *System.Speech* als .net-Referenz gibt, also die *managed code API*. Es muss kein COM mehr eingebunden werden. Das .net-Framework kann auf Windows XP, Server 2003, Vista und Server 2008 installiert werden.

Speech-Synthesizer Kodebeispiel SAPI 5.3 (C# Windows Forms)

```
public static void
SpeakText(string text, int volume, string voicename)
{
    SpeechSynthesizer ttsSynth = new SpeechSynthesizer();
    ttsSynth.SelectVoice(voicename); ttsSynth.Volume = volume;
    ttsSynth.SpeakCompleted += new
        EventHandler<SpeakCompletedEventArgs>(ttsSynth_SpeakCompleted);
    ttsSynth.SpeakAsync(text); // speak asynchron and continue with code
    // ttsSynth.Speak(text); // speak synchron and wait in code
}
static void
ttsSynth_SpeakCompleted(object sender, SpeakCompletedEventArgs e)
{
    // fertig gesprochen
}
public static void SpeakText(string text, int volume, string voicename)
{
    SpeechSynthesizer ttsSynth = new SpeechSynthesizer();
    ttsSynth.SelectVoice(voicename);
    ttsSynth.Volume = volume;
    ttsSynth.SpeakCompleted += new
        EventHandler<SpeakCompletedEventArgs>(ttsSynth_SpeakCompleted);
}
public static string
SaveToFile(string text, int volume, string voicename)
{
    SpeechSynthesizer ttsSynth = new SpeechSynthesizer();
    System.Windows.Forms.SaveFileDialog dialog = new SaveFileDialog();
    dialog.Filter = "Wave Files (*.wav)|*.wav";
    if (dialog.ShowDialog() == DialogResult.OK)
    {
        ttsSynth.SetOutputToWaveFile(dialog.FileName);
    }
}
```

```

ttsSynth.SelectVoice(voicename);
ttsSynth.Volume = volume; ttsSynth.Speak(text);
}
return dialog.FileName;
}
public static string VoiceInfo()
{
    StringBuilder text = new StringBuilder();
    foreach (InstalledVoice ttsVoice in _ttsSynth.GetInstalledVoices())
    {
        text.Append(string.Format("Name: {0}\r\n", ttsVoice.VoiceInfo.Name));
        text.Append("=====\r\n");
        text.Append(string.Format("Desc:\t{0}\r\n", ttsVoice.VoiceInfo.Description));
        text.Append(string.Format("Id:\t{0}\r\n", ttsVoice.VoiceInfo.Id));
        text.Append(string.Format("Gender:\t{0}\r\n", ttsVoice.VoiceInfo.Gender));
        text.Append(string.Format("Age:\t{0}\r\n", ttsVoice.VoiceInfo.Age));
        text.Append(string.Format("Culture:\t{0}\r\n", ttsVoice.VoiceInfo.Culture));
        foreach (SpeechAudioFormatInfo audioFormat in
            ttsVoice.VoiceInfo.SupportedAudioFormats)
        {
            text.Append(string.Format("Avg Bytes/sec:\t{0}\r\n",
                audioFormat.AverageBytesPerSecond));
            text.Append(string.Format("Bits/sec:\t{0}\r\n",
                audioFormat.BitsPerSample));
            text.Append(string.Format("BlockAlign:\t{0}\r\n",
                audioFormat.BlockAlign));
            text.Append(string.Format("ChannelCount:\t{0}\r\n",
                audioFormat.ChannelCount));
            text.Append(string.Format("EncodingFormat:\t{0}\r\n",
                audioFormat.EncodingFormat));
            text.Append(string.Format("Samples/sec:\t{0}\r\n",
                audioFormat.SamplesPerSecond));
        }
        foreach (KeyValuePair<string, string> kvp in
            ttsVoice.VoiceInfo.AdditionalInfo)
        {
            text.Append(string.Format("\t{0}: {1}\r\n", kvp.Key, kvp.Value));
        }
        text.Append("\r\n"); text.Append("\r\n"); text.Append("\r\n");
    }
    return text.ToString();
}

```

```

_recognizer.SpeechRecognized += new
    EventHandler<SpeechRecognizedEventArgs>
        (_recognizer_SpeechRecognized);
_recognizer.SpeechHypothesized += new
    EventHandler<SpeechHypothesizedEventArgs>
        (_recognizer_SpeechHypothesized);
}
private void TurnSpeechRecognitionOn()
{
    _recognizer.RecognizeAsync(RecognizeMode.Multiple);
}
void _recognizer_SpeechHypothesized
    (object sender, SpeechHypothesizedEventArgs e)
{
    Console.WriteLine(e.Result.Text); // erkannter Wortlaut
}
void _recognizer_SpeechRecognized
    (object sender, SpeechRecognizedEventArgs e)
{
    Console.WriteLine(e.Result.Text);
    // erkannter Wortlaut, ein oder mehrere Varianten
}
private SpeechSynthesizer _speechSynthesizer;
private SpeechRecognitionEngine _recognizer;
private DictationGrammar _dictationGrammar;

```

SAPI Stimmen (Voices)

Für jede eingesetzte Sprache gibt es eine sprachspezifische Stimme. D.h. für einen deutschen Text wähle ich eine Stimme, die in den Culture Eigenschaften Deutsch stehen hat. Das Mischen zwischen englischem Text und deutscher Ausgabe (oder umgekehrt) liefert zwar keinen Fehler, hört sich aber an wie „*English for runaways*“.

SAPI5

Microsoft Sam: Stimme in SAPI 5 Lernout & Hauspie: Michael und Michelle: mit Office XP und Office 2003

SAPI5.1

Microsoft Mike and Mary

Microsoft Vista bietet die Stimme Anna, diese ersetzt Microsoft Sam. Anna hört sich höherwertiger an als Sam.

Ausgelesene Beschreibung der installierten Stimmen mittels SAPI:

```

Name: Microsoft Sam
=====
Desc: Microsoft Sam
Id: MSSam
Age: Adult
Culture: en-US // englische Stimme
Gender: Male
Language: 409;9
Name: Microsoft Sam
Age: Adult
Vendor: Microsoft

```

```

Name: ScanSoft Steffi_Dri40_16kHz
=====
Desc: ScanSoft Steffi_Dri40_16kHz
Id: ScanSoftSteffi_Dri40_16kHz
Gender: Female
Age: NotSet
Culture: deDE // deutsche Stimme
Language: 407
Name: ScanSoft Steffi_Dri40_16kHz
Vendor: ScanSoft, Inc

```

Alternative sprachspezifische Stimmen

Es gibt lizenzfreie und kommerzielle Versionen sprachspezifische SAPI 5 Stimmen, hier einige Anbieter:

Anbieter	Link
Lernout & Hauspie	http://www.lhs.com/
Nuance RealSpeak Voices	http://www.nuance.com/
AT&T Natural Voices	http://www.naturalvoices.att.com/
Acapela Voices	http://www.acapela-group.com/
Cepstral	http://cepstral.com/

Sprachbestimmte Stimmen gibt es unter anderem für: American und British English, Dutch, French, German, Italien, Japanese, Korean, Portuguese, Russian, Spanish, ...

SpeechRecognizerEngine



Die Spracherkennungsoptionen aus den Systemeinstellungen von Windows Vista. (bei Windows 7 Build 7100 schaut der Dialog gleich aus)

Vor der sinnvollen Nutzung der Spracherkennung gilt es einige Arbeiten zu erledigen.

Mikrofon einrichten

Falls mehrere Mikrofone am Rechner installiert sind (Webcam könnte neben Head Set auch eines eingebaut haben), empfiehlt es sich, nur eines davon zu aktivieren und die Empfangslautstärke anzupassen.

Anschließend eine Sprechprobe machen und auf den angezeigten Pegel achten. Mikrofon zurechtrücken und Empfangslautstärke anpassen.

Computer für Spracherkennung trainieren

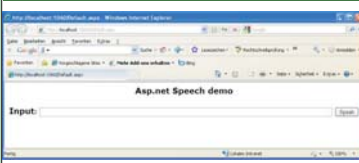
Jeder Mensch spricht (nuschelt) unterschiedlich gut, diese Unterschiede in der menschlichen Ausgabe müssen der Maschine beigebracht werden. Als Training müssen nun vorgegebene Sätze nachgesprochen werden, mit jedem Training wird die Erkennung besser. Auch ein hochwertiges Mikrofon das im richtigen Abstand vom Mund positioniert ist, trägt wesentlich zu der Erkennungsquote bei.

```

private void InitSpeech()
{
    _speechSynthesizer = new SpeechSynthesizer();
    InitializeSpeechRecognitionEngine();
    _dictationGrammar = new DictationGrammar();
}
private void InitializeSpeechRecognitionEngine()
{
    _recognizer.SetInputToDefaultAudioDevice();
    Grammar customGrammar = CreateCustomGrammar();
    _recognizer.UnloadAllGrammars();
    _recognizer.LoadGrammar(customGrammar);
}

```

Speech enabled Web Pages



Asp.net Sample mit SAPI
 In der Page Seite (aspx) muss Async true eingestellt werden:

```
// <%@ Page Language="C#"
Async="true" ...
```

 Sonst gibt es den Fehler: "Asynchronous operations are not allowed in this context. Page starting an asynchronous operation has to have the Async attribute set to true and an asynchronous operation can only be started on a page prior to PreRenderComplete event"

```
protected void btnSpeak_Click(object sender, EventArgs e)
{
    SpeechSynthesizer ttsSynth = new SpeechSynthesizer();
    ttsSynth.SpeakCompleted += new
    EventHandler<SpeakCompletedEventArgs>(ttsSynth_SpeakCompleted);
    ttsSynth.SpeakAsync(txtTextToSpeak.Text);
}
void
ttsSynth_SpeakCompleted(object sender, SpeakCompletedEventArgs e)
{
    Console.WriteLine("done");
}
```

Microsoft Speech Server

Die in der Microsoft Speech Application SDK Version 1.1 (SASDK) enthaltenen Web Controls ermöglichen eine Sprachein- und Ausgabe. Dabei wird auch die Internettelefonie durch die enthaltenen Speech Engine Services des Microsoft Speech Server unterstützt. Nach der Installation der SASDK kann in der Referenz eines Visual Studio .NET Projects das `microsoft.web.ui.speechcontrols` angegeben werden. (liegt hier: "C:\Program Files\Microsoft .NET Speech\SpeechControls\v1.0.2826.0\Microsoft.Web.UI.SpeechControls.d11".)

SALT -VoiceXML

Salt (*Speech Application Language Tags*) ist eine XML basierende Sprache, die in HTML und XHTML Seiten verwendet werden kann, um damit in Web Seiten eine Sprachausgabe zu ermöglichen. Salt Forum wurde im Jahr 2001 von Microsoft, Cisco Systems, Comverse, Intel, Philips consumer electronics und Scansoft gegründet. Die Salt ist bei W3C eingereicht worden und 2002 abgenommen.

Im Jahr 2004 hat W3C hat VoiceXML als Standard empfohlen. Daher unterstützt der Microsoft Speech Server 2004 Salt, die Version 2007 unterstützt Salt und VoiceXML 2.0 und 2.1.

Für den Microsoft Internet Explorer gibt es ein Speech Plug In, der Salt unterstützt.

VoiceXML ist ebenfalls XML basierend. Es wurde von AT&T, IBM, Lucent und Motorola entwickelt.

Die aktuelle Version ist 2.1, die zukünftige Version 3.0 unterstützt dann SCXML (*State chart XML*), um damit auch den Status einer *State Machine* beschreiben zu können.

VoiceXML verbindet Internet und Telefonie, indem es mit der XML-Sprache der Webseite sowohl die Sprachausgabe als auch die Erkennung unterstützt.

VoiceXML Elements

Element	Beschreibung
<audio>	Gibt den Synthesize Modus für text to speech oder eine Audio Datei an
<form>	Gibt den Beginn eines VXML files an.
<field>	Variable Feld: Bsp true/false oder Zahlen
<prompt>	Wartet auf Eingabe des Benutzers
<noinput>	Dieser Event kann nach einem Prompt kommen, und zwar dann, wenn der Benutzer keinen Input liefert oder akustisch nicht gehört werden kann. Dabei kann der Count Parameter genutzt werden, was in den einzelnen erfolgreichen Events zu machen oder zu sprechen ist.
<nomatch>	Event der Eintritt, wenn die Benutzereingabe nicht den vorgegebenen Regeln entspricht.
<help>	Der Benutzer benötigt Hilfe.
<filled>	Event wenn eine gültige Eingabe getätigt wurde.
<catch>	Fehlermechanismus
<goto>	Navigation zwischen den Forms oder zu anderen VXML Dokumenten

<submit> Übertragen des Inhalts des Forms zu einer externen Quelle mit HTTP GET oder POST.

VoiceXML Entwicklungs Tools

Anbieter	Link
Audium	http://www.audiumcorp.com/
BeVocal	http://cafe.bevocal.com/
Clarity	http://www.clarity-ag.de/
Entervoice	http://www.entervoice.com/
HeyAnita	http://www.heyanita.com/
Idylic	http://www.idylic.com/
Invores	http://www.invores.com/
Microsoft Speech API Microsoft Office	http://msdn.microsoft.com/en-us/library/ms723627(VS.85).aspx
Communications Server 2007 Speech Server	http://msdn.microsoft.com/en-us/library/bb857803.aspx
Pipebeach (HP OpenCall Media Platform)	http://h20208.www2.hp.com/openCall/products/media/ocmp/index.jsp
Plum Voice	http://www.plumvoice.com/
SkyCreek	http://www.skycreek.com/
Speaklink	http://www.speaklink.com/
Tellme	https://studio.tellme.com/
Vocomo Software	http://searchportal.information.com/
VoiceObjects	http://www.voiceobjects.com/
VoiceGenie	http://www.genesyslab.com/products/self_service.asp
VoiceShot	http://www.voiceshot.com/
Vox Tiger	http://www.voxtiger.com/
Voxeo	http://www.voxeo.com/
Voxpilot	http://ode.voxpilot.com

Zusammenfassung

Sprachausgaben eines Computers durch synthetische Stimmen werden heute eingesetzt, wo man eine dynamische datenaktuelle Sprachausgabe benötigt. Also etwa auf einen Flughafen, wo Fluggäste über Aktivitäten rund um Landungen, Verspätungen, Personensuchmeldungen etc. mittels einer Durchsage informiert werden. Die Ansagen werden hintereinander in mehreren Sprachen ausgegeben, der Tonfall der synthetischen Stimme ist dabei immer der selbe, man merkt dies. Die nicht sprachspezifischen Namen der Fluggäste (also nicht in der *Culture* der eingesetzten Stimme) werden ziemlich witzig ausgesprochen. Zugegeben auch nicht besser wie es der menschliche Sprecher macht. Als Gegenbeispiel eines fix aufgesprochenen Textes in einer einzigen Sprache sind die Ansagen in Bus und Strassenbahn. Wenn etwas außerplanmäßiges eintritt, muss die Durchsage direkt erfolgen: „Zuuuuuuuugg fäääh-Rumps“ in der sehr menschlichen *“Culture“* der Stimme.

So könnte eine Sprachausgabe / Erkennung noch sinnvoll genutzt werden:

Eine Beispielanwendung wäre eine telefonische Auskunft einer Supportanfrage. Nach Wahl einer Nummer werden mir Texte vorgesprochen, nach der ich eine Entscheidung mittels Nummer oder Sprach-eingabe treffen kann.

Computerstimme: „Drücken Sie die #2, wenn Sie ein Gerät des Typs xy haben“.

Dahinter wird ein Workflow abgearbeitet, bei dem der Text vorgelesen wird.

Ich drücke die 2.

Computerstimme: „Bitte Auftragsnummer eingeben oder sprechen“.

Ich spreche 12345.

Nach gesprochener Auftragsnummer fragt die Computerstimme nach:

Computerstimme: „Ihre Auftragsnummer lautet 12345, ist das korrekt?“

Die Sprachausgabe kann aufgrund der abgefragten Daten generiert werden.

Computerstimme: „Ihre Reparatur des Geräts xyz vom 1.7.2009 wurde durchgeführt, die Servicekosten betragen 80 Euro. Sie können das Gerät am 12.7.2009 abholen“.