SNOBOL4 Web Service Wrapper

Making the classic SNOBOL4 programming language accessible through modern web browsers

Overview

This project creates a web-based interface for the SNOBOL4 programming language (circa 1962). Instead of requiring users to install SNOBOL4 locally, this service provides a REST API and web interface that allows anyone to execute SNOBOL4 code through their browser.

Live Demo

🚀 **Try it online:** https://snobolserver.onrender.com

Architecture

The system consists of three main components:

```
[Web Browser] ↔ [Node.js Server] ↔ [SNOBOL4 Engine]
```

- **Frontend (index.html)**: Simple web interface with textarea for code input
- **Backend (server.js)**: Express.js server that handles API requests
- **SNOBOL4 Engine**: Compiled SNOBOL4 interpreter from source code

Data Flow

1. User writes SNOBOL4 code in web interface
2. JavaScript sends code via HTTP POST to `/run-snobol` endpoint
3. Server writes code to temporary file
4. Server spawns SNOBOL4 process to execute the file
5. Server captures output/errors and returns JSON response
6. Web interface displays results

Installation & Setup

Option 1: Run Locally

Prerequisites:

- Node.js 18+ and npm

- Linux/Unix environment (for SNOBOL4 compilation)

Steps:

```bash
# Clone the repository
git clone https://github.com/yourusername/snobol-server.git
cd snobol-server

# Install dependencies
npm install

# Compile SNOBOL4 (Linux/Unix only)
wget https://ftp.regressive.org/snobol4/snobol4-2.3.3.tar.gz
tar -xvf snobol4-2.3.3.tar.gz
cd snobol4-2.3.3
./configure && make
sudo cp snobol4 /usr/local/bin/snobol4
cd .. && rm -rf snobol4-2.3.3*

# Update server.js to point to system binary
# Change: const snobolExecutable = path.join(__dirname, 'snobol4');
# To: const snobolExecutable = '/usr/local/bin/snobol4';

# Start the server
node server.js
```

Access at: `http://localhost:3000`

Option 2: Docker (Recommended)

Prerequisites:

- Docker installed

Steps:

```bash
# Clone the repository
git clone https://github.com/yourusername/snobol-server.git
cd snobol-server

# Build Docker image
docker build -t snobol4-server .

# Run container
docker run -p 3000:3000 snobol4-server
```

Access at: `http://localhost:3000`

Option 3: Deploy to Render (Cloud)

Prerequisites:

- GitHub account
- Render account (free tier available)

Steps:

1. **Prepare your repository:**

   ```bash
   # Make sure your Dockerfile is in the root directory
   # Update server.js to use: const snobolExecutable = '/usr/local/bin/snobol4';

   git add .
   git commit -m "Prepare for Render deployment"
   git push origin main
   ```

2. **Deploy on Render:**
   - Go to render.com and sign up/login
   - Click "New +" → "Web Service"
   - Connect your GitHub repository
   - Render will automatically detect the Dockerfile
   - Choose "Docker" as the environment
   - Set Build Command: `docker build`
   - Set Start Command: `docker run`

    - Click "Deploy"

3. **Access your service:**
    - Render provides a URL like: `https://yourapp.onrender.com`
    - First deployment takes 5-10 minutes (SNOBOL4 compilation)
    - Subsequent deployments are faster

Note for Render Free Tier: Service may spin down after 15 minutes of inactivity, causing a 50+ second delay on first request.

API Reference

Execute SNOBOL4 Code

Endpoint: `POST /run-snobol`

Headers:

```
Content-Type: application/json
```

Request Body:

```json
{
  "code": "    OUTPUT = 'Hello World!'\nEND"
}
```

Response (Success):

```json
{
  "output": "Hello World!\n",
  "error": "",
  "exitCode": 0
}
```

Response (Error):

```json
{
  "output": "",
  "error": "SNOBOL4 syntax error details...",
  "exitCode": 1
}
```

JavaScript Example

```javascript
async function runSnobol(code) {
    const response = await fetch('/run-snobol', {
        method: 'POST',
        headers: { 'Content-Type': 'application/json' },
        body: JSON.stringify({ code })
    });

    const result = await response.json();
    console.log('Output:', result.output);
    if (result.error) console.error('Error:', result.error);
    return result;
}

// Usage
runSnobol(`
    TEXT = "Hello SNOBOL4!"
    OUTPUT = TEXT
END`);
```

SNOBOL4 Programming Tips

SNOBOL4 has specific syntax requirements:

1. **Always end with `END`:**

    ```snobol
    OUTPUT = "Hello World!"
    END
    ```

2. **Use proper indentation (spaces or tabs):**

```snobol
    TEXT = "The cat sat on the mat"
    TEXT = REPLACE(TEXT,"cat","dog")
    OUTPUT = TEXT
END
```

3. **Pattern matching example:**

```snobol
    TEXT = "SNOBOL4 is fun"
    TEXT "SNOBOL4" = "Programming"
    OUTPUT = TEXT
END
```

File Structure

```
snobol-server/
├── Dockerfile              # Docker container configuration
├── package.json            # Node.js dependencies
├── server.js              # Express.js web server
├── index.html            # Web interface
└── README.md             # This file
```

Technical Notes

- **SNOBOL4 Compilation**: The Dockerfile compiles SNOBOL4 from source to ensure compatibility with the container environment
- **Security**: Each execution creates a temporary file, runs in isolated process, and cleans up automatically
- **Error Handling**: Captures both stdout and stderr from SNOBOL4 process
- **File Cleanup**: Temporary files are automatically deleted after execution

Troubleshooting

"GLIBC version not found" error:

- This happens when using a pre-compiled binary on a different system
- Solution: Use the Docker approach which compiles SNOBOL4 in the target environment

"Command not found" error:

- Check that `snobolExecutable` path in server.js points to the correct location
- For local install: `/usr/local/bin/snobol4`
- For Docker: `/usr/local/bin/snobol4`

SNOBOL4 syntax errors:

- Remember to end programs with `END`
- Use proper indentation
- Check SNOBOL4 documentation for language syntax

Contributing

This project is designed for SNOBOL4 enthusiasts who want to share the language with others. Contributions welcome:

- Bug fixes and improvements
- Better error handling
- Additional SNOBOL4 examples
- Documentation improvements

License

This project is open source. The SNOBOL4 interpreter is distributed under its original license terms.

About SNOBOL4

SNOBOL4 (String Oriented and Symbolic Language) was developed at Bell Labs in 1962. It excels at:

- String manipulation and pattern matching
- Text processing and parsing
- Symbolic computation
- Teaching programming language concepts

Despite its age, SNOBOL4 remains relevant for understanding pattern matching concepts that influence modern languages.

Made with ❤ for the SNOBOL4 community