Cheat Sheet

Basic Usage

```
from vcon import Vcon
from vcon.party import Party
from vcon.dialog import Dialog
```

Creating vCons

```
# Create new vCon
vcon = Vcon.build_new()

# Create from dictionary
vcon = Vcon({"uuid": "...", "vcon": "0.0.1"})

# Create from JSON
vcon = Vcon.build_from_json(json_string)
```

Serialization

```
# To JSON
json_str = vcon.to_json()  # or vcon.dumps()

# To dictionary
dict_data = vcon.to_dict()
```

Parties

Creating & Adding Parties

```python
# Create party
party = Party(
    tel="+1234567890",
    name="John Doe",
    mailto="john@example.com",
    role="customer",
    meta={"custom": "data"}
)

# Add to vCon
vcon.add_party(party)

# Find party
index = vcon.find_party_index("name", "John Doe")
```

Dialog

Supported MIME Types

- Text: `text/plain`
- Audio: `audio/x-wav`, `audio/wav`, `audio/mpeg`, `audio/mp3`, `audio/ogg`, `audio/webm`
- Video: `video/x-mp4`, `video/ogg`

Adding Dialog

```python
# Text dialog
dialog = Dialog(
    type="text",
    start="2024-03-21T10:00:00Z",
    parties=[0, 1],
    mimetype="text/plain",
    body="Hello, how can I help?",
    originator=0
)
vcon.add_dialog(dialog)

# Audio dialog with external URL
audio_dialog = Dialog(
    type="recording",
    start="2024-03-21T10:01:00Z",
    parties=[0, 1],
    url="http://example.com/audio.wav",
    mimetype="audio/wav",
    duration=120.5
)
vcon.add_dialog(audio_dialog)
```

Dialog Methods

```python
# Add inline data
dialog.add_inline_data(
    body="content",
    filename="message.txt",
    mimetype="text/plain"
)

# Add external data
dialog.add_external_data(
    url="http://example.com/file.wav",
    filename="recording.wav",
    mimetype="audio/wav"
)

# Convert external to inline
dialog.to_inline_data()
```

Attachments & Analysis

Attachments

```
# Add attachment
vcon.add_attachment(
    type="transcript",
    body="Content here",
    encoding="none"  # Options: "none", "base64", "base64url"
)

# Find attachment
attachment = vcon.find_attachment_by_type("transcript")
```

Analysis

```
# Add analysis
vcon.add_analysis(
    type="sentiment",
    dialog=[0],  # Dialog indices
    vendor="analyzer",
    body={"score": 0.8},
    encoding="json",  # Options: "json", "none", "base64url"
)

# Find analysis
analysis = vcon.find_analysis_by_type("sentiment")
```

Tags & Metadata

```
# Add tag
vcon.add_tag("category", "support")

# Get tag
value = vcon.get_tag("category")

# Access all tags
tags = vcon.tags
```

Security & Validation

Signing & Verification

```
# Generate keys
private_key, public_key = Vcon.generate_key_pair()

# Sign vCon
vcon.sign(private_key)

# Verify signature
is_valid = vcon.verify(public_key)
```

Validation

```
# Validate object
is_valid, errors = vcon.is_valid()

# Validate file
is_valid, errors = Vcon.validate_file("conversation.json")

# Validate JSON string
is_valid, errors = Vcon.validate_json(json_string)
```

UUID Generation

```
# From domain name
uuid = Vcon.uuid8_domain_name("example.com")

# With custom bits
uuid = Vcon.uuid8_time(custom_bits)
```

Properties Reference

Vcon Properties

- `uuid`: Unique identifier
- `vcon`: Version number

- `created_at`: Creation timestamp
- `updated_at`: Last update timestamp
- `parties`: List of participants
- `dialog`: List of dialog entries
- `attachments`: List of attachments
- `analysis`: List of analysis entries
- `redacted`: Redaction information
- `group`: Group information
- `meta`: Metadata

Party Properties

- `tel`: Telephone number
- `stir`: STIR verification
- `mailto`: Email address
- `name`: Party name
- `validation`: Validation status
- `gmlpos`: Geographic position
- `civicaddress`: Civic address
- `uuid`: Unique identifier
- `role`: Party role
- `meta`: Additional metadata