Friday the JQ'th

Json Lives

Slashing APIs with jq

Ulrik Aabye-Hansen

Interim camp counselor & trainer

Macysadmin 2025

Ulrik Aabye-Hansen

Trainer & Consultant

WARNING

This presentation contains references to murder, mutilation, and violent imagery — all within a fictional, horror-movie context.

Any depictions of blood, violence, or victims are purely comedic and satirical, drawing only from the world of slasher films.

Please be advised if you are sensitive to such themes, even when presented humorously.

disclaimer

This presentation makes use of references, imagery, and characters inspired by popular horror films, including the Friday the 13th series.

References are employed strictly for parody and educational purposes, and are not intended as copyright infringement on any copyrighted material.

All characters and names remain the property of their respective rights holders.

Agenda

Alone in the cabin

A terminal condition

What even is JSON

Inside the Camp

String it up

jq - your chainsaw for JSON

Leaving the Camp

Venturing on to APIs

Alone in the cabin

A "terminal" condition

1980

awk

pattern-directed scanning
and processing language

1984

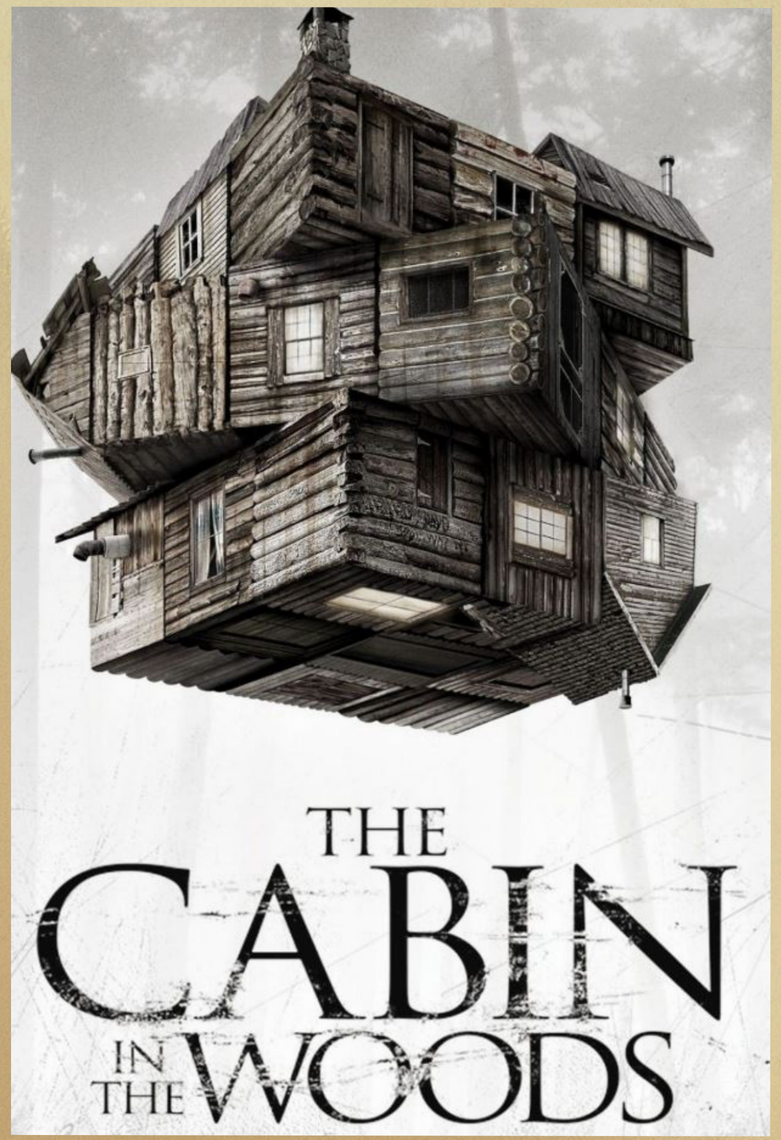
grep

file pattern searcher

1999

xml

Extended markup language



2012

jq

JSON query

What even IS JSON?

Other than, "that guy"

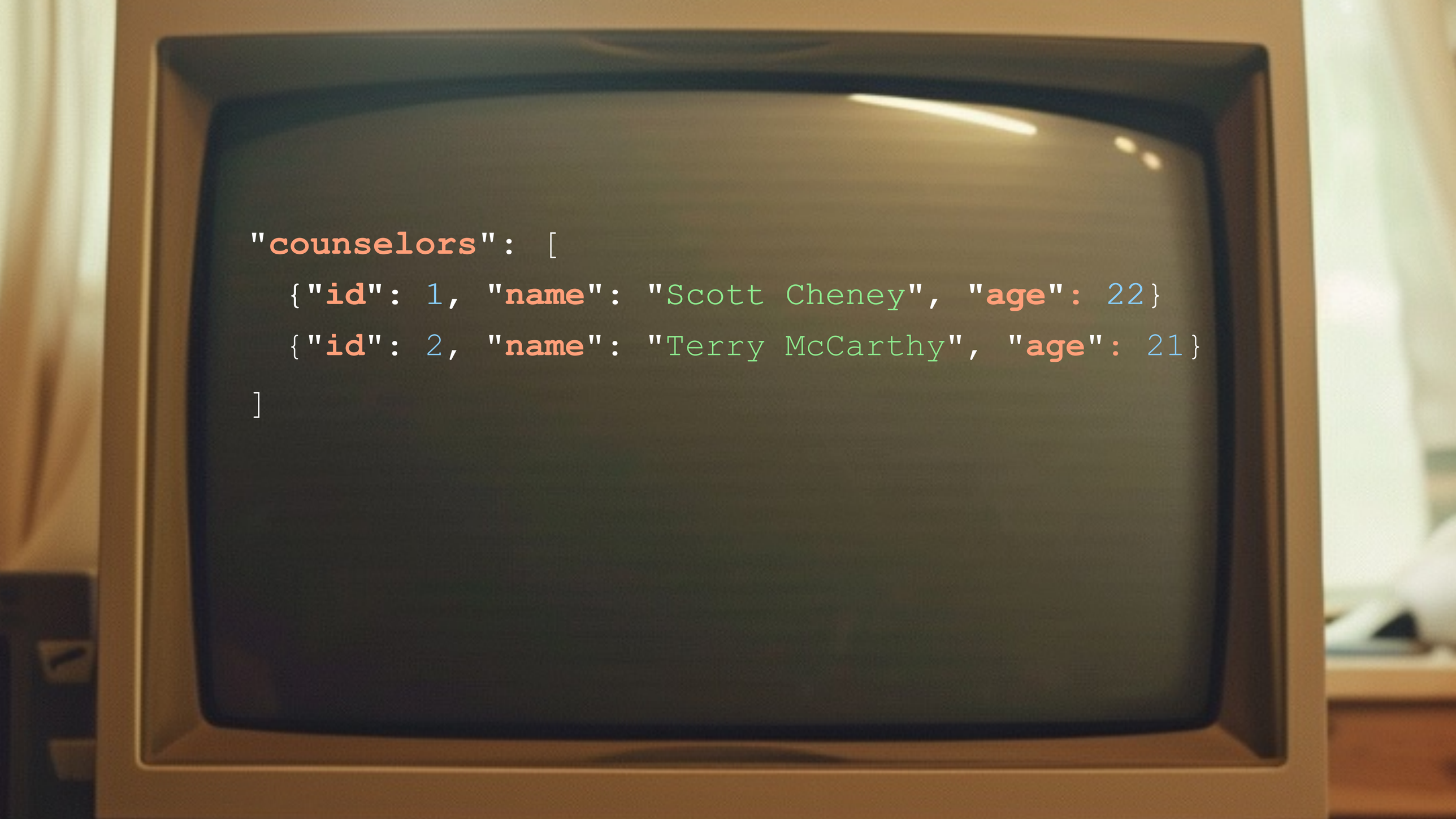

CAMP
LOG


{
"year": 1981,
"headCounselor": "Paul Holt",
"finalApproval": true,
"location": "Crystal Lake"
"cancellations": null
}

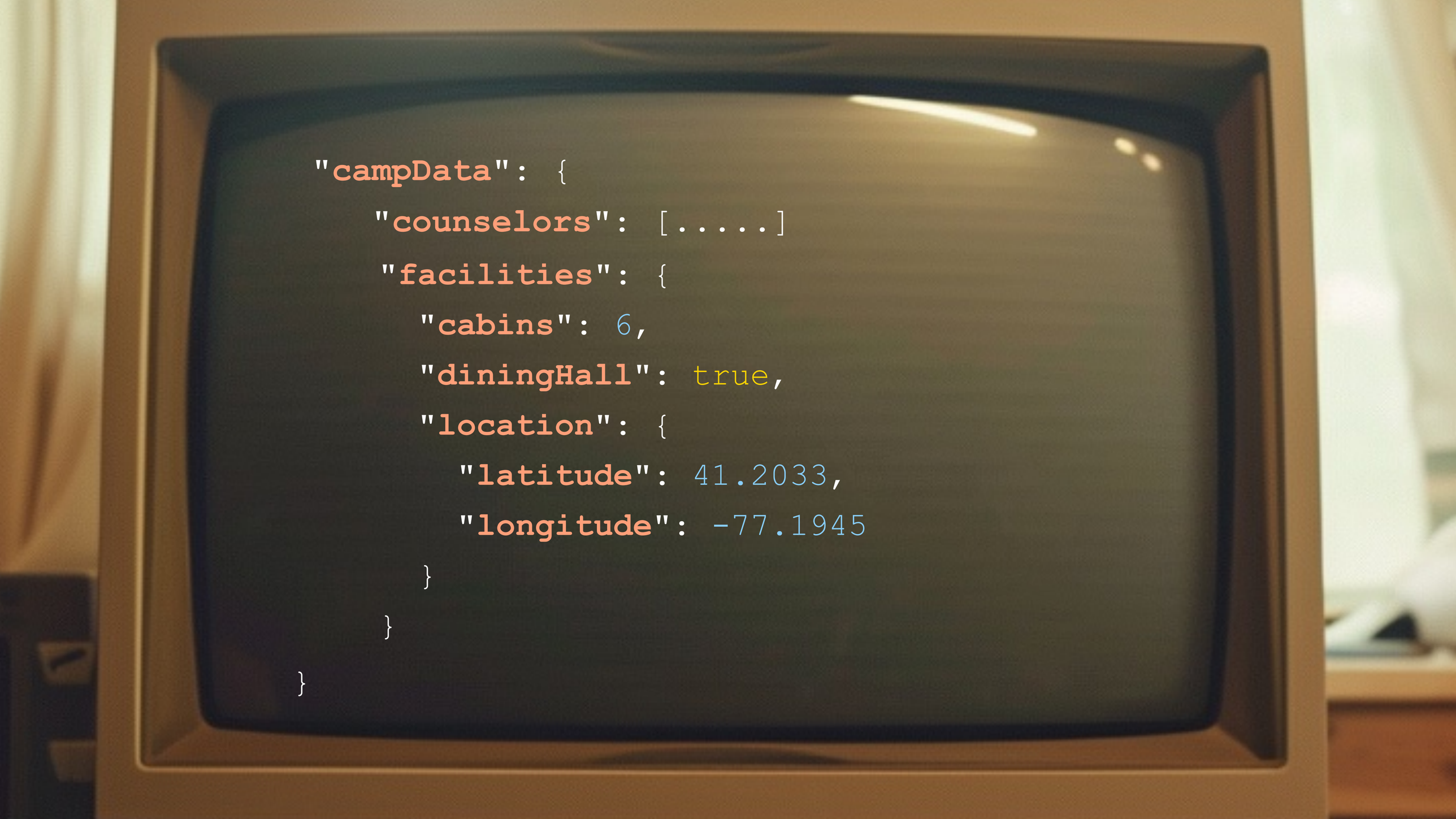
JSON Data Types & Examples

Element	Example	Notes
🔑 Keys (attributes)	"HeadCounselor"	key in quotes with :
🔤 Strings	"Crystal Lake"	Text in **double quotes**
🔢 Numbers	1981	digits - **no quotes**
✅ Boolean (true/false)	true	all lowercase - **no quotes**
🕳️ Null	null	all lowercase - **no quotes**


"campData": {
"counselors": [.....]
}


"counselors": [
{"id": 1, "name": "Scott Cheney", "age": 22}
{"id": 2, "name": "Terry McCarthy", "age": 21}
]


"counselors": [
{"id": 1,
"name": "Scott Cheney",
"age": 22,
"shift": "evening",
"relationships": {
"romanticInterest": "Terry McCarthy"
}
}
]


"campData": {
"counselors": [.....]
"facilities": {
"cabins": 6,
"diningHall": true,
"location": {
"latitude": 41.2033,
"longitude": -77.1945
}
}
}

From camp-data to API

JSON is a common language of APIs

Used by **Apple Business Manager**, **Jamf Pro**, **Intune**, ++

Understanding the structure is the first step to parsing with `jq`

Slicing it up

`jq` - your chainsaw for JSON

Meet: `jq`

`jq` = “JSON Query”

CLI tool for slicing, filtering, and transforming JSON

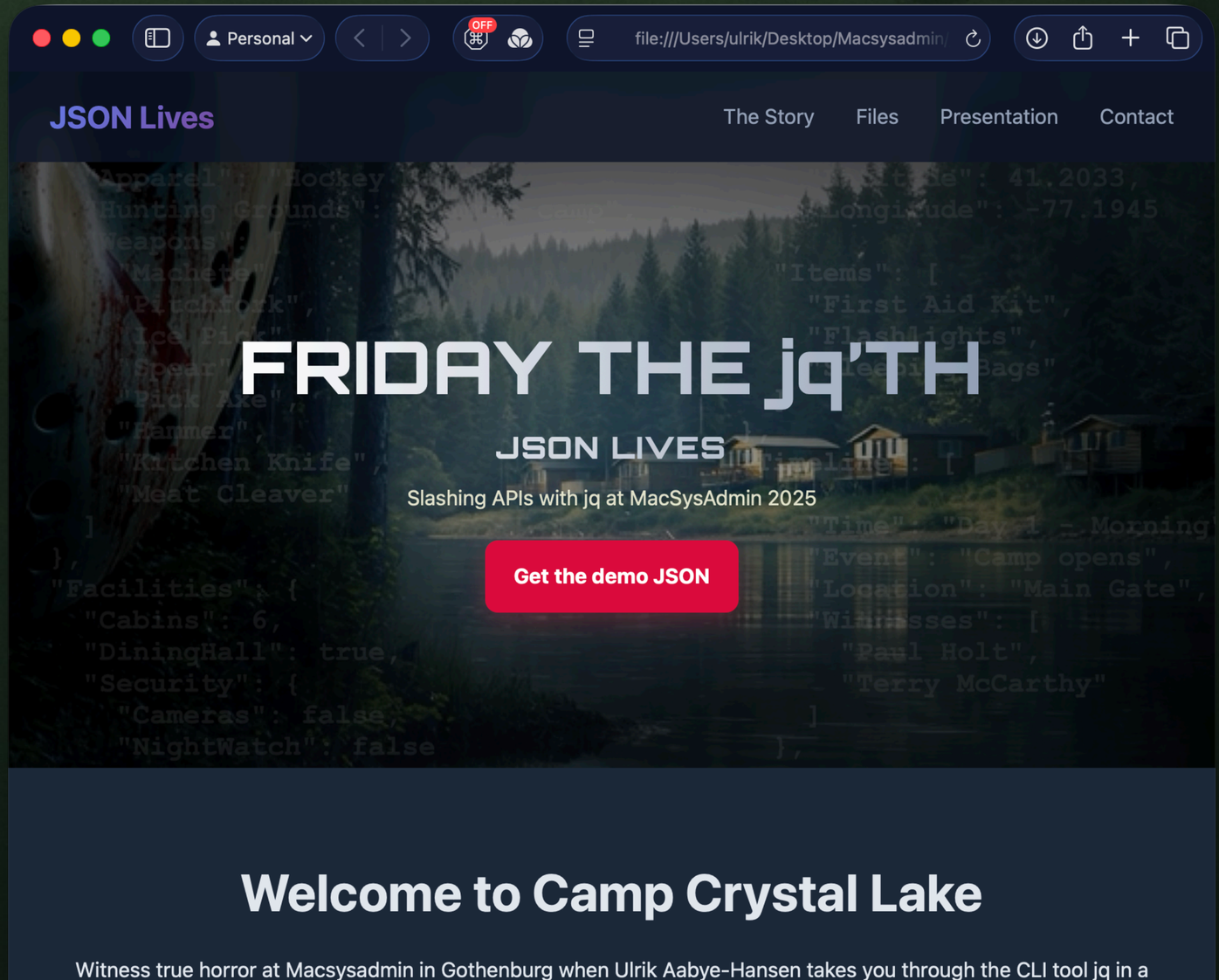
Like grep or awk for **structured data**

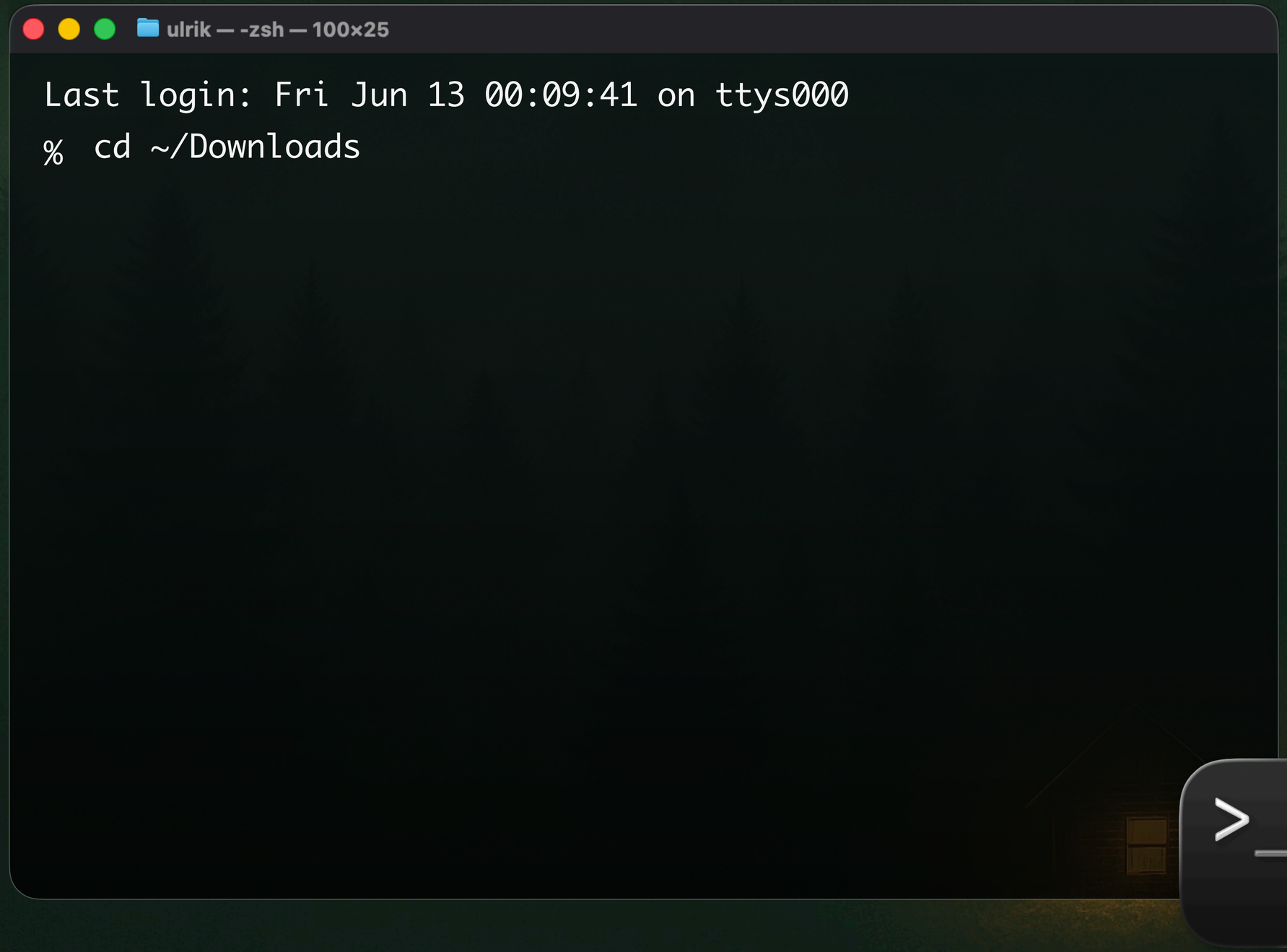
Minimal syntax

jsonlives.com

Download JSON file to follow along demos

Download presentation and additional demo files later today




ulrik — -zsh — 100×25
Last login: Fri Jun 13 00:09:41 on ttys000
% cd ~/Downloads

camp_crystal_lake.json

```
{
  "year": 1981,
  "headCounselor": "Paul Holt",
  "finalApproval": true,
  "location": "Crystal Lake"
  "cancellations": null
  "campData": { ... },
  "crystalLakeKiller": { ... }}
}
```

ulrik — -zsh — 45×20

```
Last login: Fri Jun 13 00:09:41 on ttys000
%  jq '.year' camp_crystal_lake.json
```

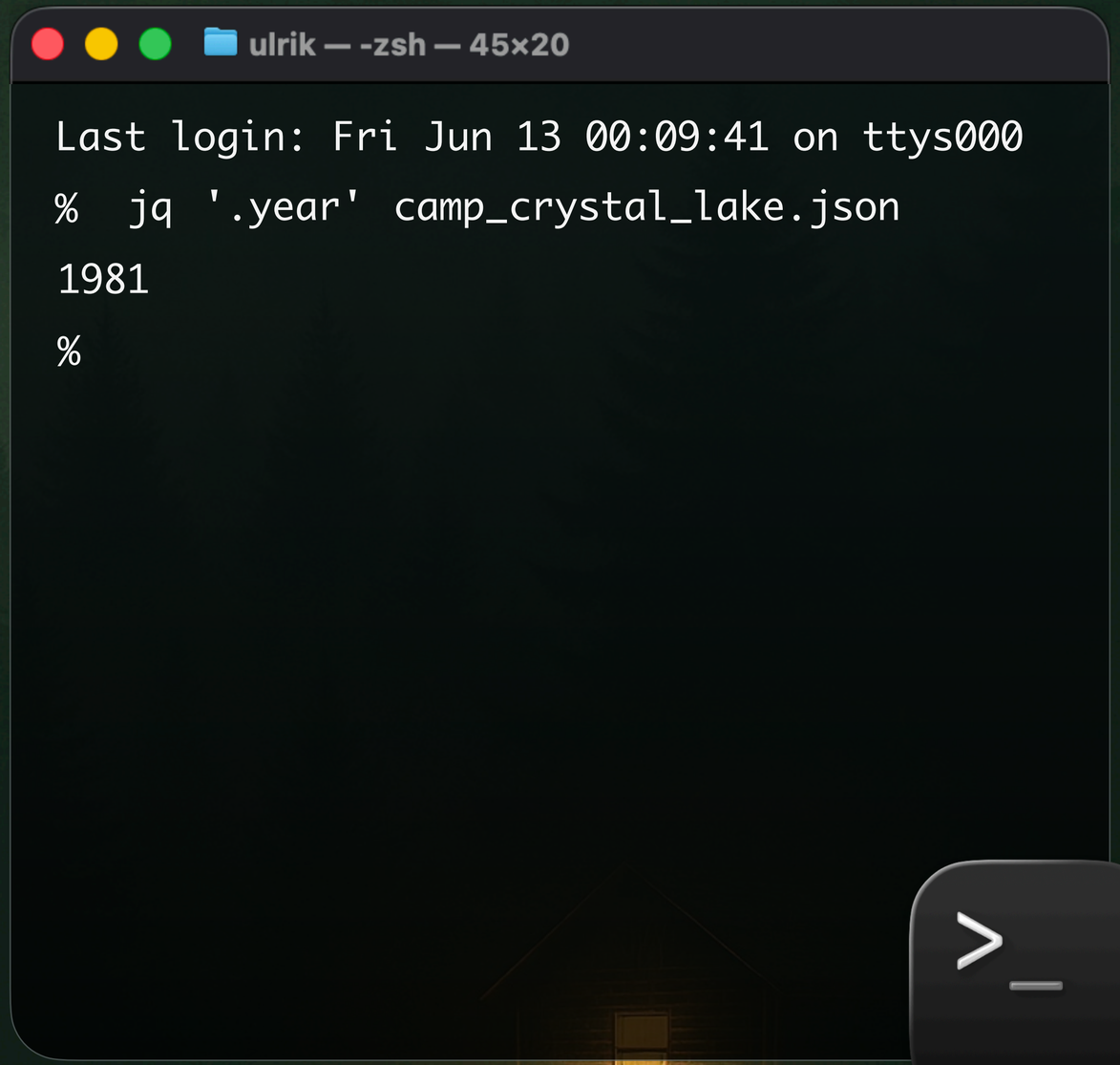
command

filter

file

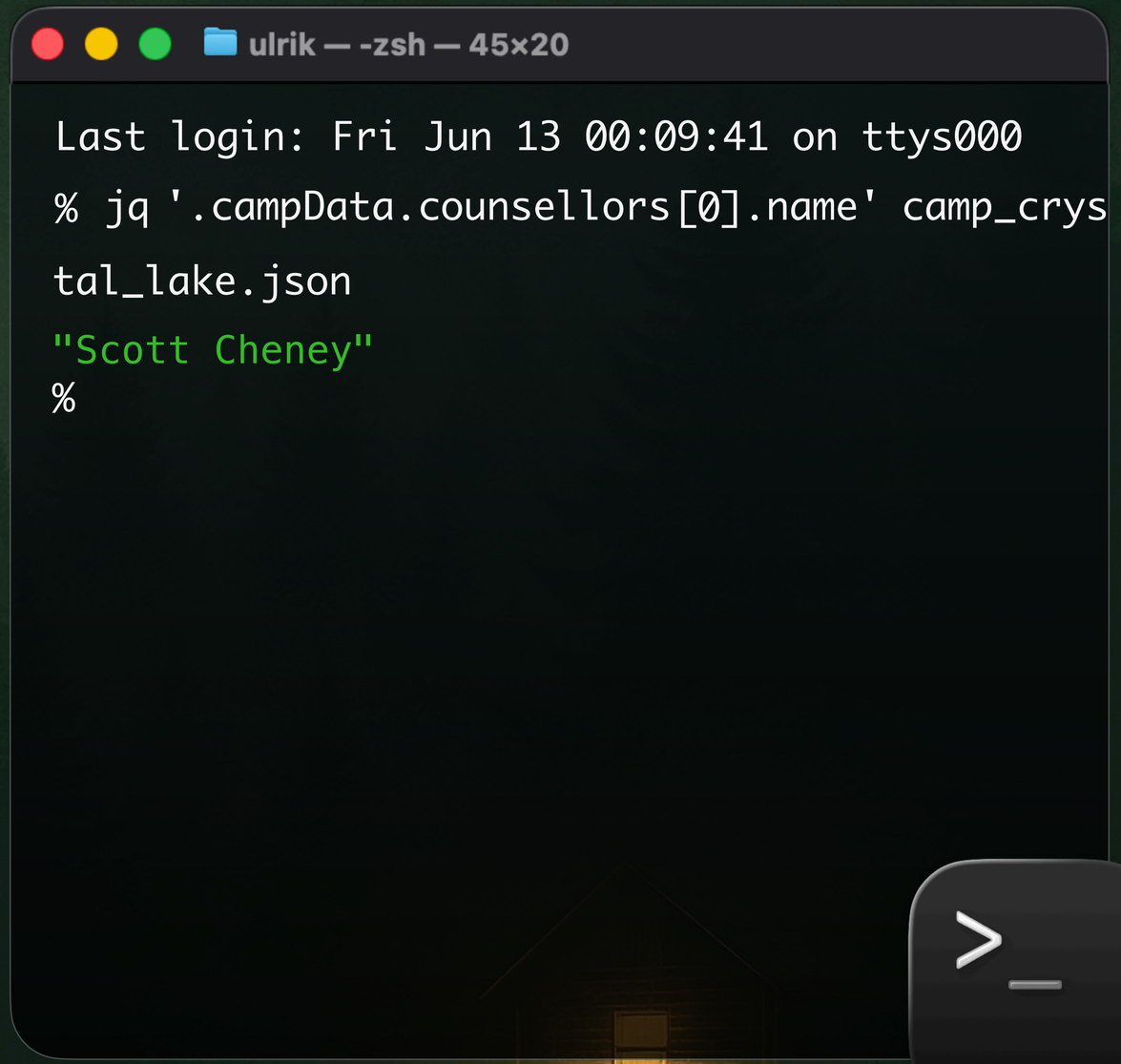
camp_crystal_lake.json

```
{
  "year": 1981,
  "headCounselor": "Paul Holt",
  "finalApproval": true,
  "location": "Crystal Lake"
  "cancellations": null
  "campData": { ... },
  "crystalLakeKiller": { ... }}
}
```



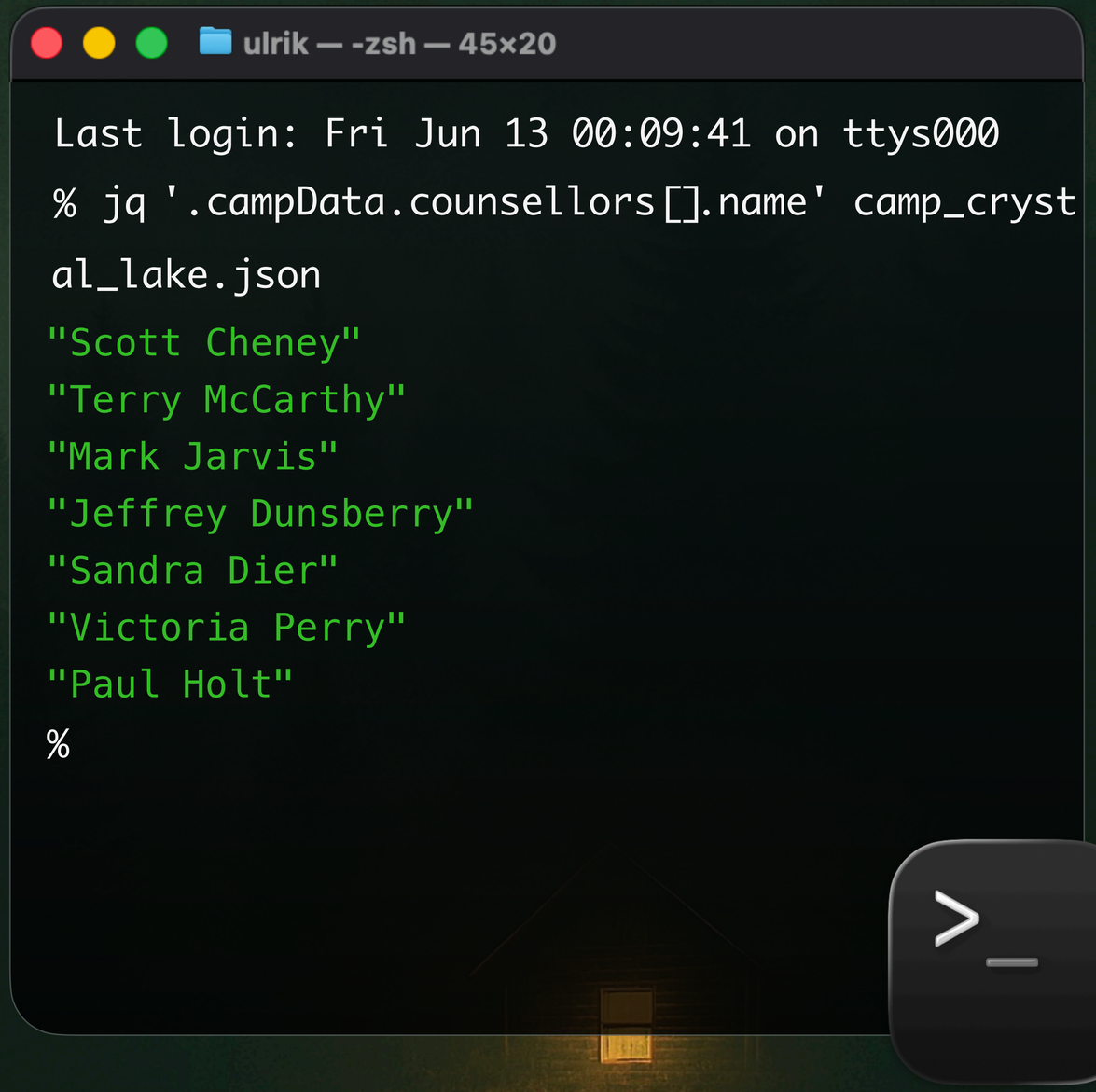
camp_crystal_lake.json

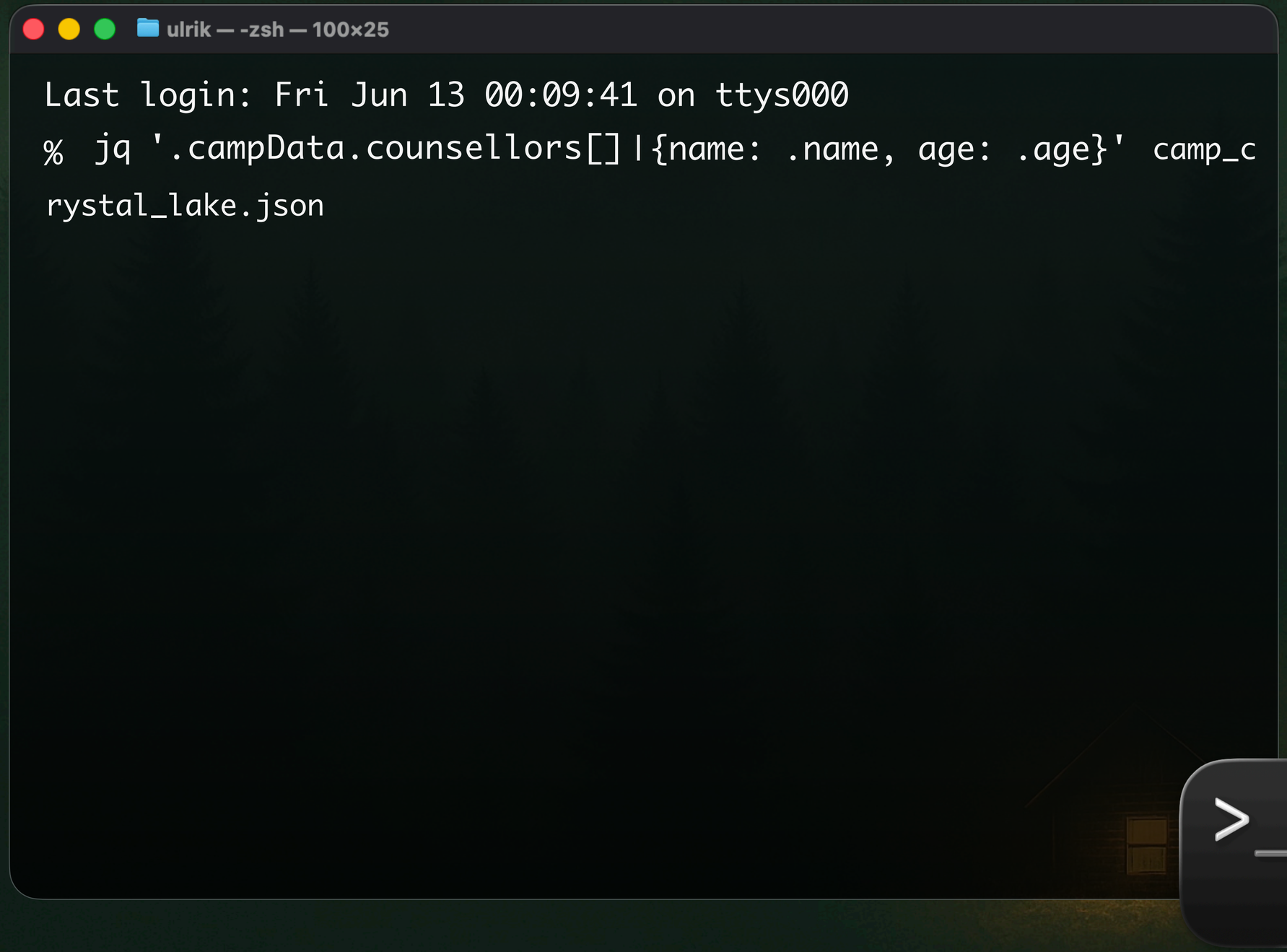
```
{ ...
 "campData": {
   "counsellors": [
     { "id": 1,
       "name": "Scott Cheney",
       "age": 22,
       "shift": "evening",
       "relationships": {
        "romanticInterest": "Terry
        McCarthy" }
   },
   ...
]
```

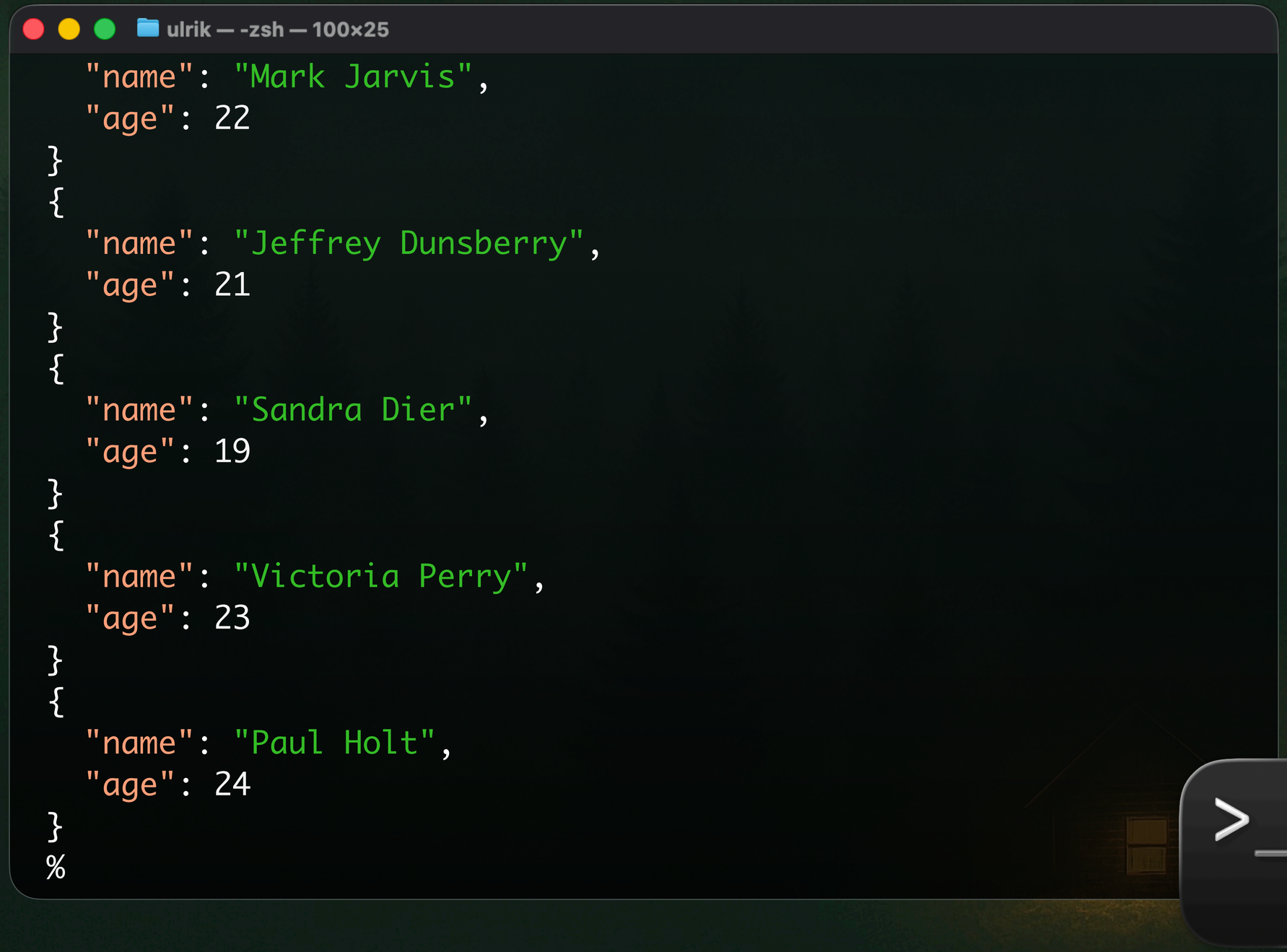


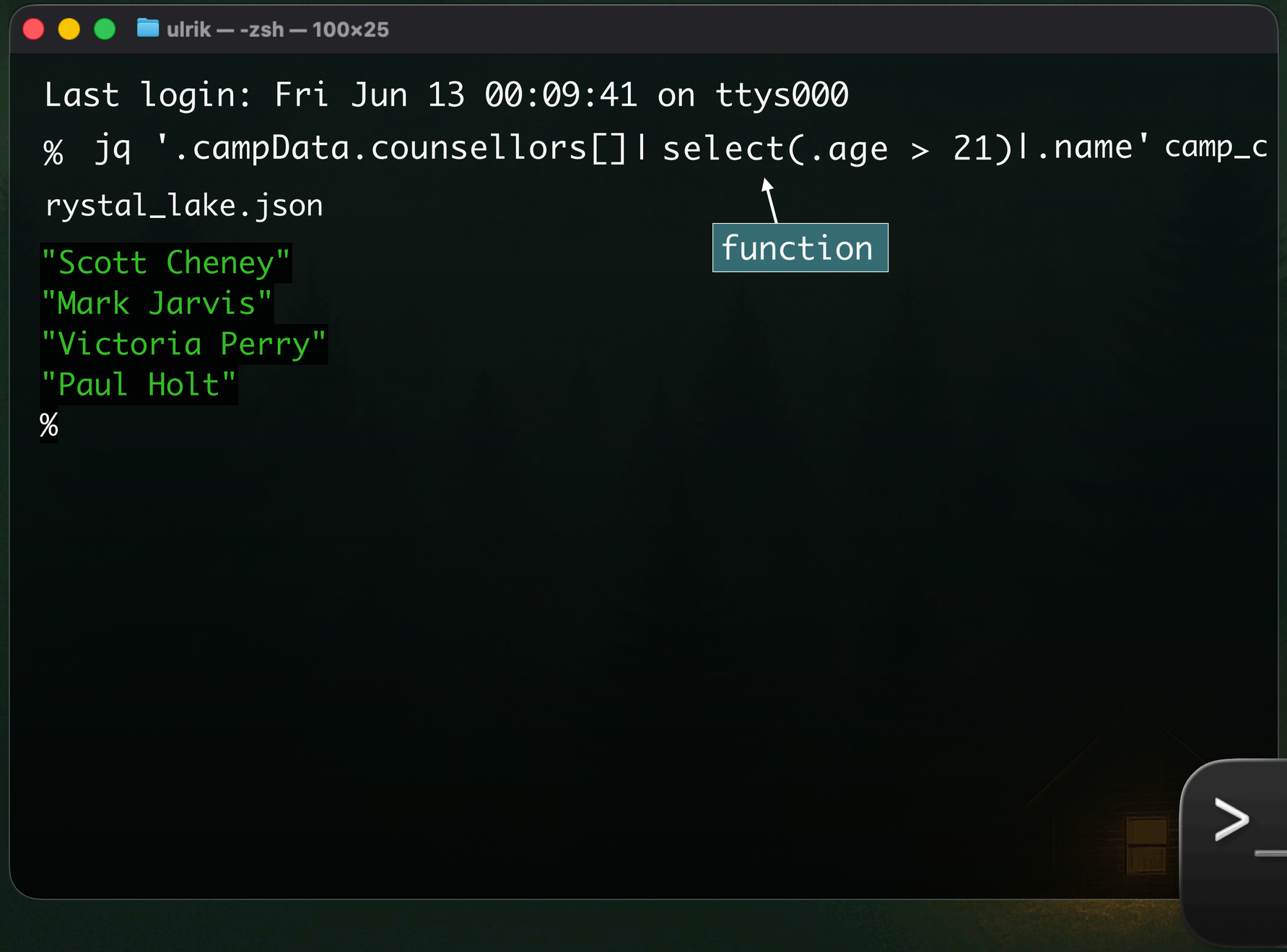
camp_crystal_lake.json

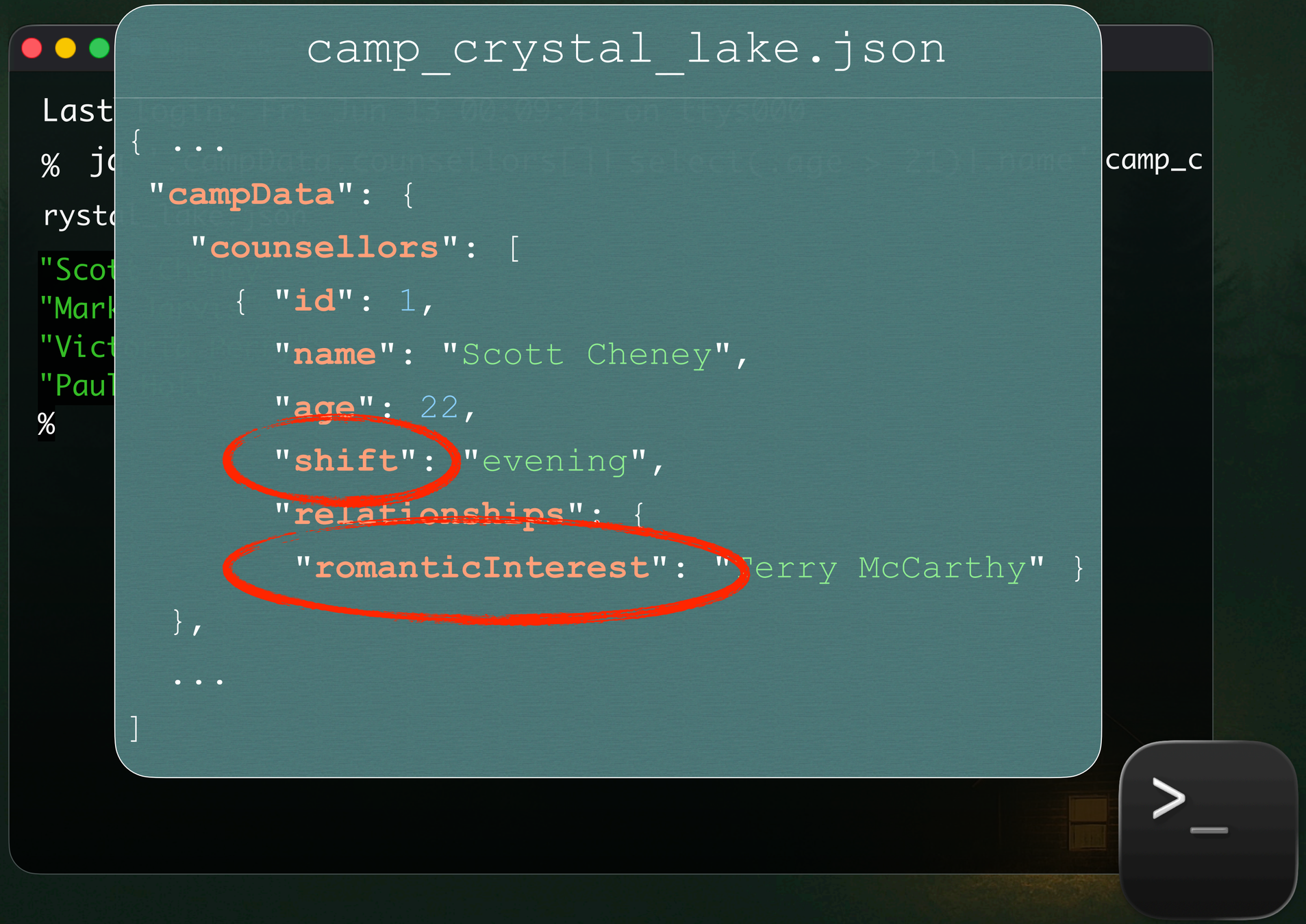
```
{ ...
 "campData": {
   "counsellors": [
     { "id": 1,
       "name": "Scott Cheney",
       "age": 22,
       "shift": "evening",
       "relationships": {
        "romanticInterest": "Terry
        McCarthy" }
  },
  ...
]
```

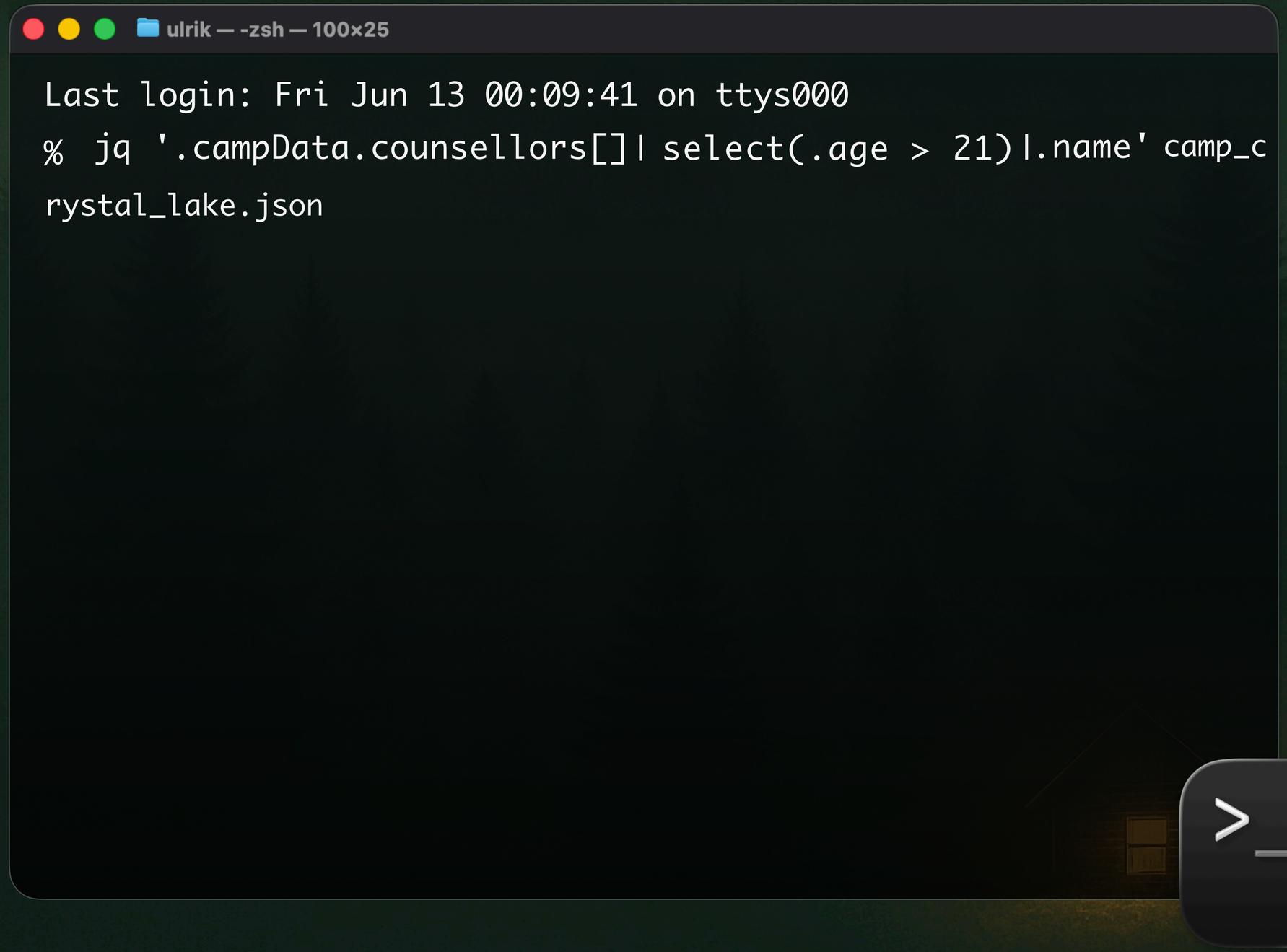


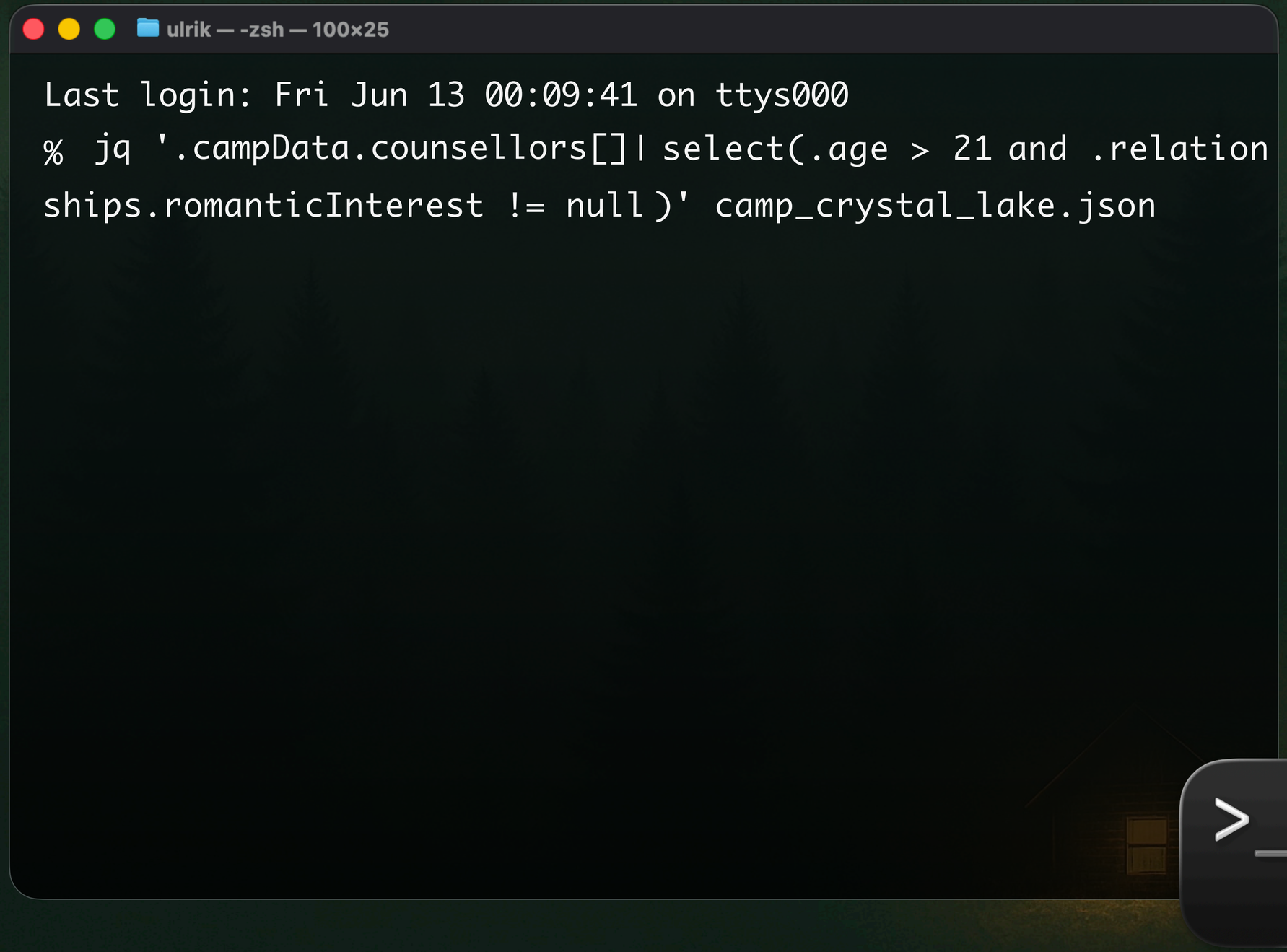

ulrik — -zsh — 100×25
Last login: Fri Jun 13 00:09:41 on ttys000
% jq '.campData.counsellors[]|{name: .name, age: .age}' camp_c
rystal_lake.json


ulrik — -zsh — 100×25
"name": "Mark Jarvis",
"age": 22
}
{
"name": "Jeffrey Dunsberry",
"age": 21
}
{
"name": "Sandra Dier",
"age": 19
}
{
"name": "Victoria Perry",
"age": 23
}
{
"name": "Paul Holt",
"age": 24
}
%

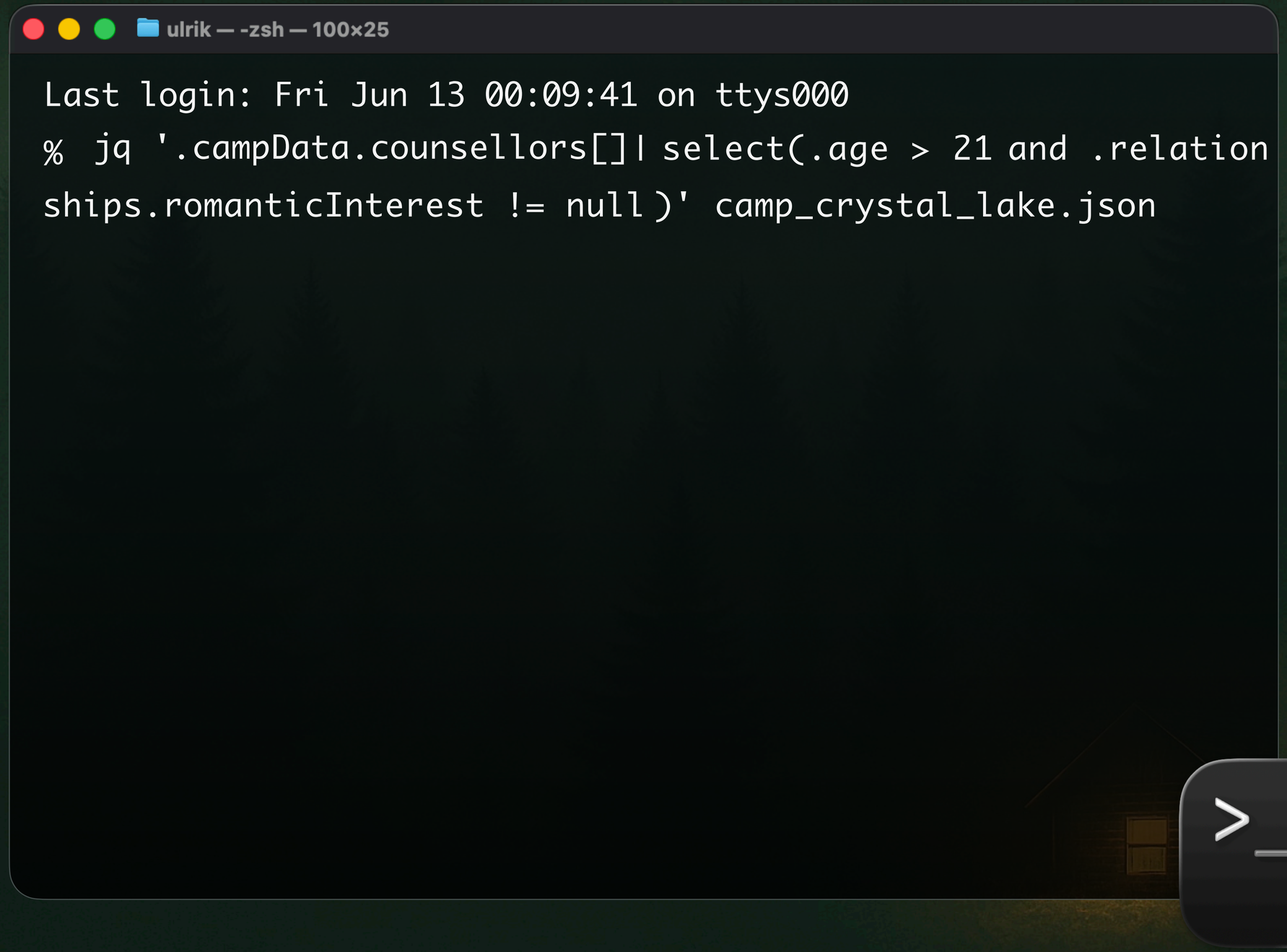

ulrik — -zsh — 100×25
Last login: Fri Jun 13 00:09:41 on ttys000
% jq '.campData.counsellors[]| select(.age > 21)|.name' camp_c
rystal_lake.json
"Scott Cheney"
"Mark Jarvis"
"Victoria Perry"
"Paul Holt"
%
function

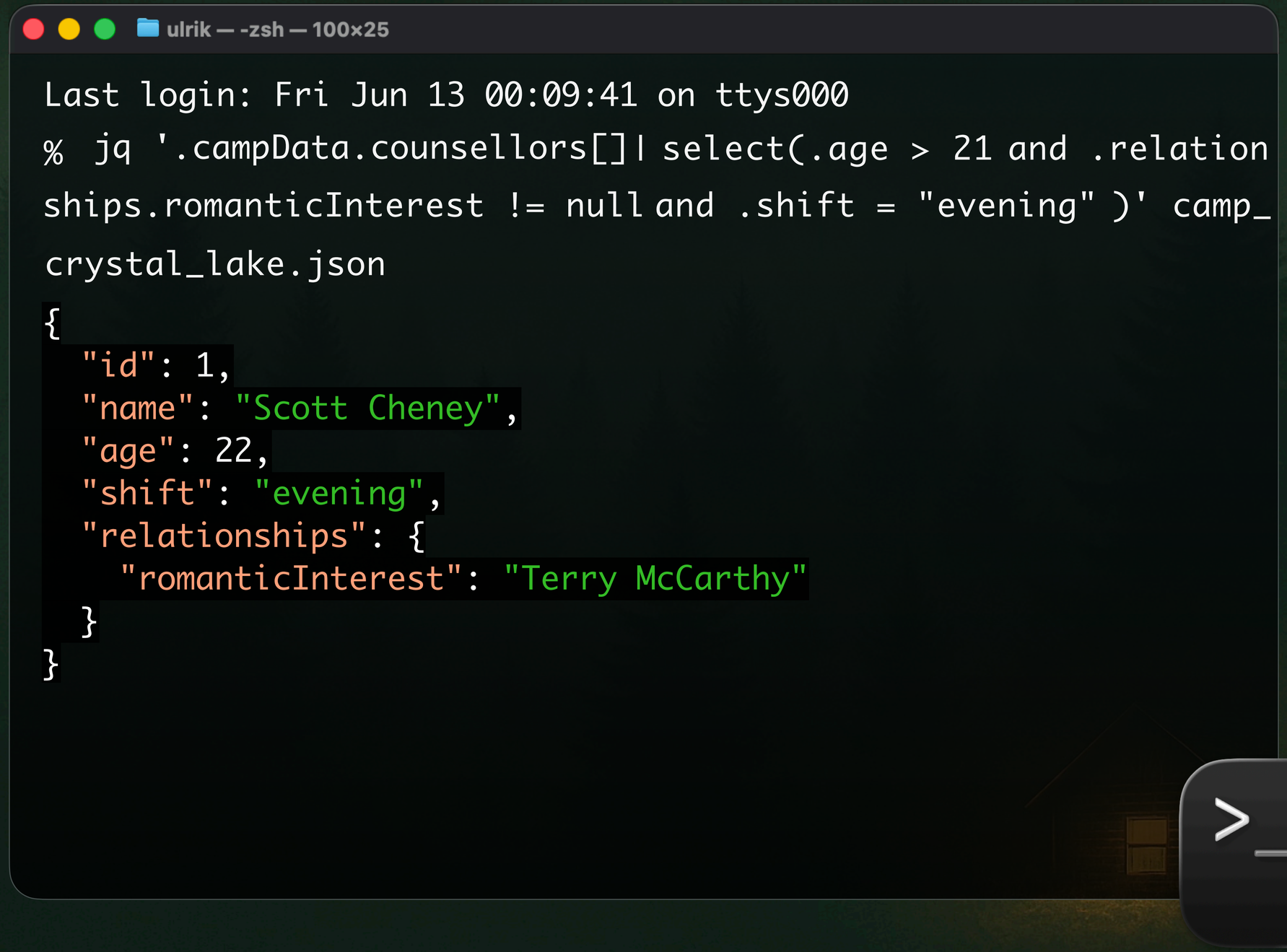

camp_crystal_lake.json
{ ...
"campData": {
"counsellors": [
{ "id": 1,
"name": "Scott Cheney",
"age": 22,
"shift": "evening",
"relationships": {
"romanticInterest": "Jerry McCarthy" }
},
...
]


ulrik — -zsh — 100×25
Last login: Fri Jun 13 00:09:41 on ttys000
% jq '.campData.counsellors[]| select(.age > 21)|.name' camp_c
rystal_lake.json


ulrik — -zsh — 100×25
Last login: Fri Jun 13 00:09:41 on ttys000
% jq '.campData.counsellors[]| select(.age > 21 and .relation
ships.romanticInterest != null)' camp_crystal_lake.json


ulrik — -zsh — 100×25
}
}
{
"id": 3,
"name": "Mark Jarvis",
"age": 22,
"shift": "afternoon",
"relationships": {
"romanticInterest": "Victoria Perry"
}
}
{
"id": 6,
"name": "Victoria Perry",
"age": 23,
"shift": "afternoon",
"relationships": {
"romanticInterest": "Mark Jarvis"
}
}
%


ulrik — -zsh — 100×25
Last login: Fri Jun 13 00:09:41 on ttys000
% jq '.campData.counsellors[]| select(.age > 21 and .relation
ships.romanticInterest != null)' camp_crystal_lake.json

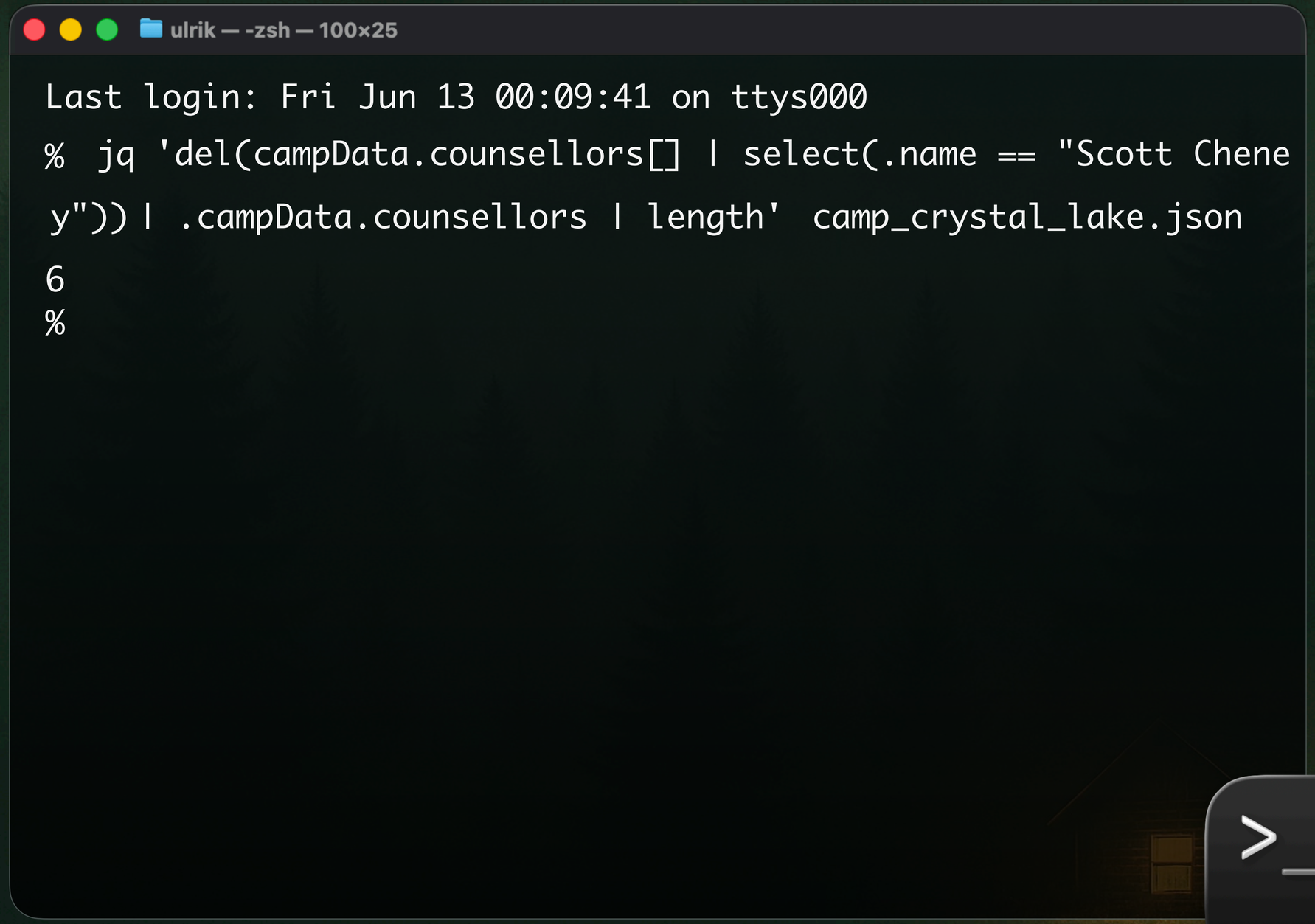

ulrik — -zsh — 100×25
Last login: Fri Jun 13 00:09:41 on ttys000
% jq '.campData.counsellors[]| select(.age > 21 and .relationships.romanticInterest != null and .shift = "evening")' camp_crystal_lake.json
{
"id": 1,
"name": "Scott Cheney",
"age": 22,
"shift": "evening",
"relationships": {
"romanticInterest": "Terry McCarthy"
}
}

camp_crystal_lake.json

```
{ ...
 "campData": {
   "counsellors": [
     { "id": 1,
       "name": "Scott Cheney",
       "age": 22,
       "shift": "evening",
       "relationships": {
        "romanticInterest": "Terry
        McCarthy" }
  },
  ...
]
```

ulrik — -zsh — 45×20

```
Last login: Fri Jun 13 00:09:41 on ttys000
% jq 'del(.campData.counsellors[] | select
(.name == "Scott Cheney"))' camp_crystal_lake.
json
{
  "year": 1981,
  "headCounselor": "Paul Holt",
  "campData": {
    "counsellors": [
      {
        "id": 2,
        "name": "Terry McCarthy",
        "age": 21,
        "shift": "morning",
        "relationships": {
          "romanticInterest": "Scott C
```


ulrik — -zsh — 100×25
Last login: Fri Jun 13 00:09:41 on ttys000
% jq 'del(campData.counsellors[] | select(.name == "Scott Cheney")) | .campData.counsellors | length' camp_crystal_lake.json
6
%

Following Jasons blood trail

Element	Example	Notes
Array Iteration	`.campData.counsellors[]`	Check every cabin
Age Filter	`select(.age > 21)`	Target experience
Null Check	`select(.romanticInterest != null)`	Filter distracted
Text Match	`select(.shift == "evening")`	Night workers
Combine Filters	`select(.age > 21 and .shift == "evening")`	Perfect victim
Remove Data	`del(select(.name == "Scott"))`	The kill
Counting	`| length`	Body count list

Leaving the camp

Venturing on to APIs

Home eating vs. restaurant

Cooking @home	Restaurent Experience
Go to grocery store	Walk into restaurant
Search every aisle for ingredients	Look at the menu
Pay for groceries	Showing your ID – reservation
Buy ingredients, drive home	Tell waiter your order
Prep, cook, season, serve	Dinner is served for you

Restaurent → API

Restaurent Experience	API Component	What It Does
`Walk into restaurant`	curl command	Your tool to make the request
`Showing your ID - reservation`	Authentication	Proving you're allowed to order
`Look at the menu`	API Endpoint	Specific objects
`Tell waiter your order`	API Request	Asking for what you want
`Dinner is served for you`	JSON Response	The data you requested

A sacrifice for the Demo-gods

A sacrifice for the Demo-gods

A sacrifice for the Demo-gods

"Oh, Demo-gods, accept this sacrifice of

🍌'Banana Girl.'🍌

Let her death not be in vain, and may the 3 demos that follow be as swift and clean as Jason's blade."

Let's get down to

Apple Business Manager

Live Demo

You can't run from JAMF Pro

Live Demo

Graph(ic) Interaction with Intune

Live Demo

Thank you

Icons