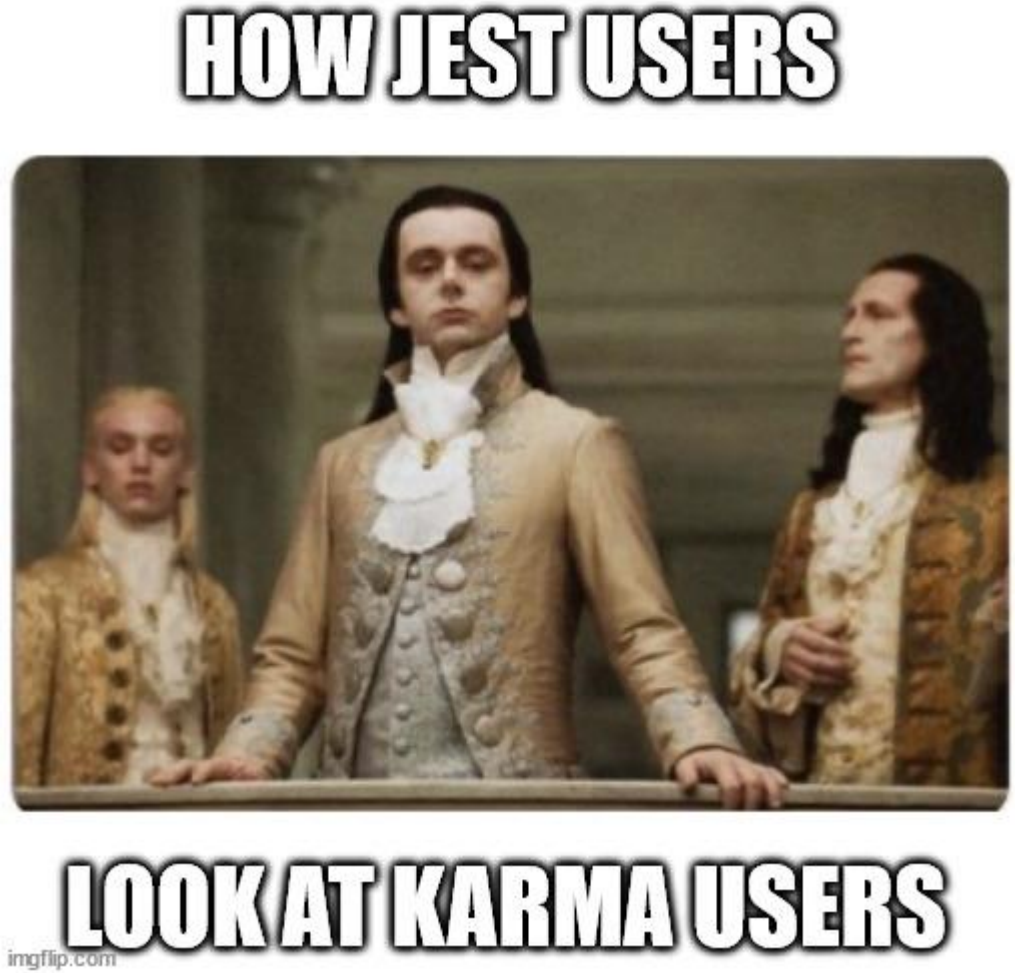
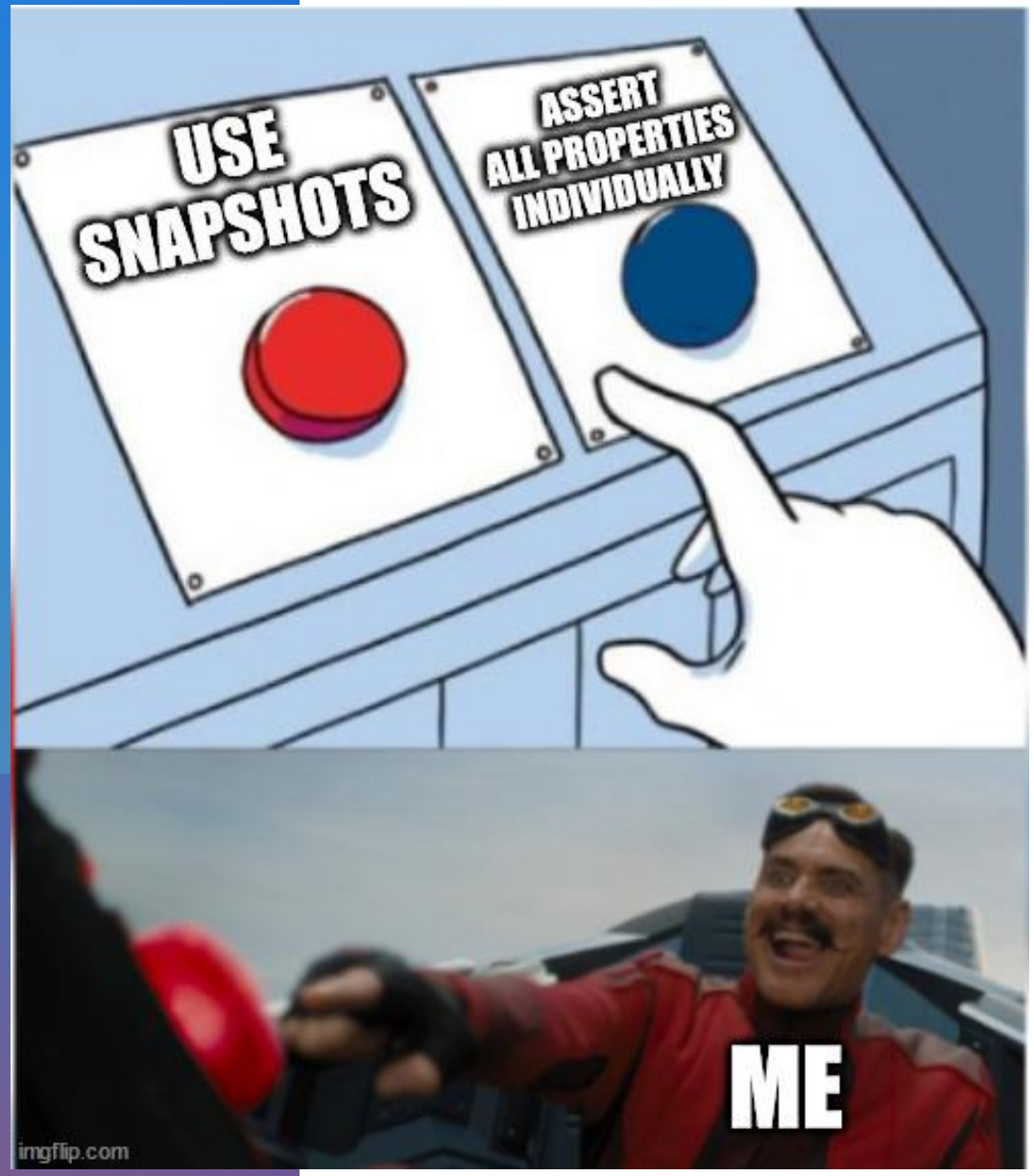
THE WAY I JEST ANGULAR APPS

MARCEL VON WYL

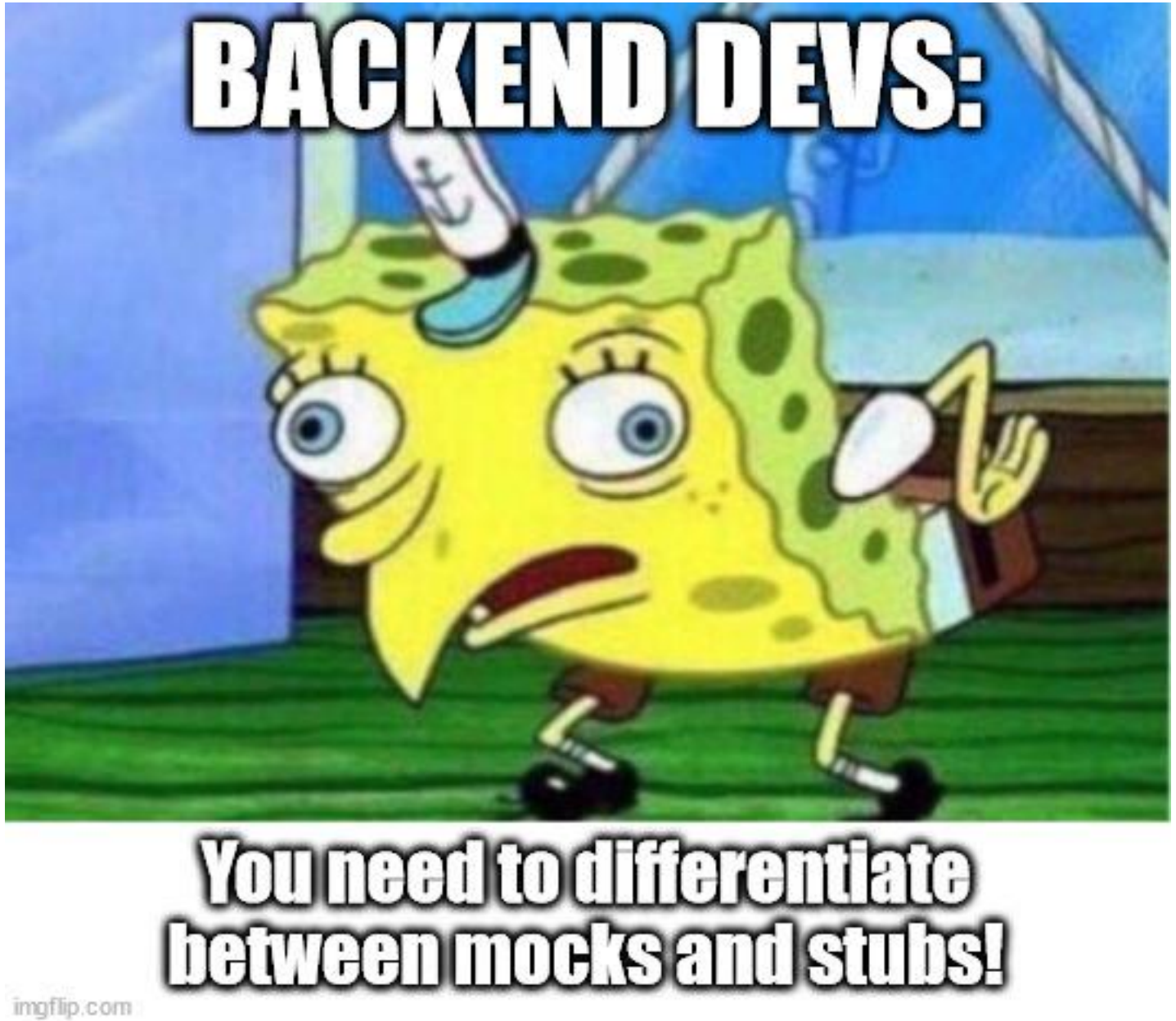
TESTING FRAMEWORKS

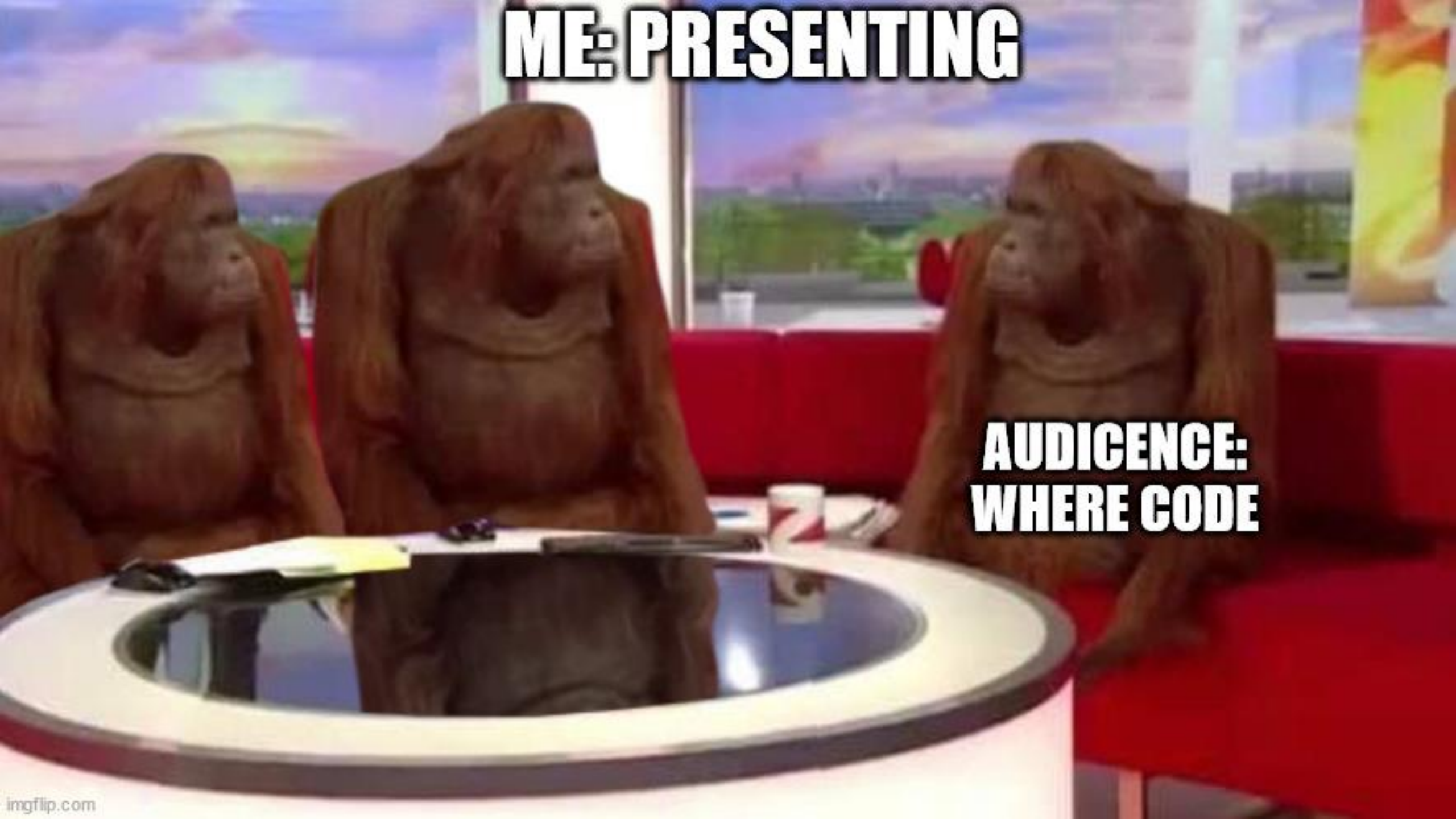


SNAPSHOTS


USE SNAPSHOTS
ASSERT ALL PROPERTIES INDIVIDUALLY
ME
imgflip.com

TEST DOUBLES

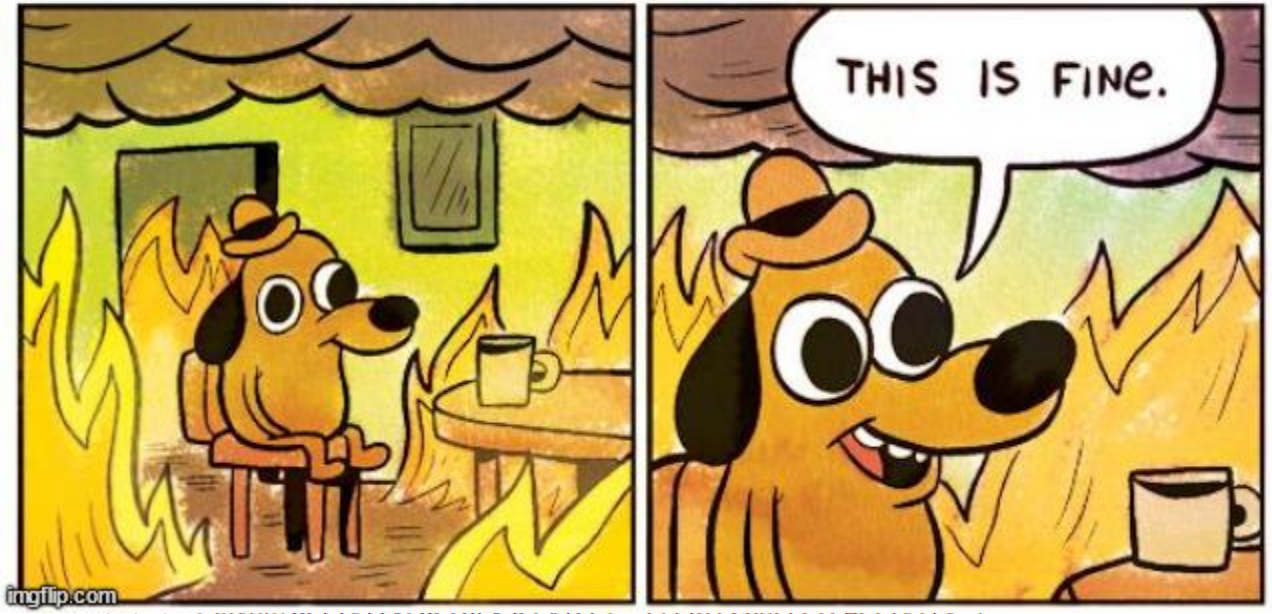



ME: PRESENTING
AUDICENCE:
WHERE CODE
imgflip.com

```
describe('ActivitiesEffects', () => {
  let actions$: Observable<Action>;
  let effects: ActivitiesEffects;
  let mockStore: MockStore;
  let streamServiceMock: StreamService;
  let documentServiceMock: DocumentService;
  let activityFactoryService: CockpitActivityFactoryService;

  const initialState = {
    [cockpitFeatureKey]: { activities: initialActivitiesState },
  };

  beforeEach(() => {
    TestBed.configureTestingModule({
      providers: [
        ActivitiesEffects,
        provideMockStore({ initialState }),
        provideMockActions(() => actions$),
        StreamService,
        DocumentService,
        CockpitActivityFactoryService,
      ],
    });

    effects = TestBed.inject(ActivitiesEffects);
    mockStore = TestBed.inject(MockStore);
    streamServiceMock = TestBed.inject(StreamService);
    documentServiceMock = TestBed.inject(DocumentService);
    activityFactoryService = TestBed.inject(CockpitActivityFactoryService);

    jest
      .spyOn(activityFactoryService, 'createConcreteActivity')
      .mockImplementation((activity) =>
        Object.assign(new MockActivity(), activity)
      );
  });
  //...
}
```

```
describe('ActivitiesEffects', () => {
  let actions$: Observable<Action>;
  let effects: ActivitiesEffects;
  let mockStore: MockStore;
  let streamServiceMock: StreamService;
  let documentServiceMock: DocumentService;
  let activityFactoryService: CockpitActivityFactoryService;

  const initialState = {
```

needing to provide all
dependencies and their dependencies




```
      .mockImplementation((activity) =>
        Object.assign(new MockActivity(), activity)
      );
  });
  //...
}
```

```
class StreamServiceMock {
  getActivityStream = jest.fn();
  otherFunction = jest.fn().mockImplementation((value: number) => value++)
}


TestBed.configureTestingModule({
  providers: [
    ActivitiesEffects,
    provideMockStore({ initialState }),
    provideMockActions(() => actions$),
    {provide: StreamService, useValue: StreamServiceMock},
    DocumentService,
    CockpitActivityFactoryService,
  ],
});
```

```
describe('ActivitiesEffects', () => {
  let actions$: Observable<Action>;
  let effects: ActivitiesEffects;
  let mockStore: MockStore;
  let streamServiceMock: StreamService;
  let documentServiceMock: DocumentService;
  let activityFactoryService: CockpitActivityFactoryService;

  const initialState = {
    [cockpitFeatureKey]: { activities: initialActivitiesState },
  };

  beforeEach(() => {
    TestBed.configureTestingModule({
      providers: [
        ActivitiesEffects,
        provideMockStore({ initialState }),
        provideMockActions(() => actions$),
        provideMock(StreamService),
        provideMock(DocumentService),
        provideMock(CockpitActivityFactoryService),
      ],
    });

    effects = TestBed.inject(ActivitiesEffects);
    mockStore = TestBed.inject(MockStore);
    streamServiceMock = TestBed.inject(StreamService);
    documentServiceMock = TestBed.inject(DocumentService);
    activityFactoryService = TestBed.inject(CockpitActivityFactoryService);

    jest
      .spyOn(activityFactoryService, 'createConcreteActivity')
      .mockImplementation((activity) =>
        Object.assign(new MockActivity(), activity)
      );
  });
  //...
}
```

```ts
it('should return activities', waitForAsync(() => {
  // arrange
  actions$ = of(ActivitiesActions.loadActivitiesRemotely());

  const activities = [
    {
      id: 'id1',
      type: CockpitActivityType.RECEIVED,
    } as Activity,
  ];

  const activityStreamSpy = jest
    .spyOn(streamServiceMock, 'getActivityStream')
    .mockReturnValue(of(activities));

  mockStore.overrideSelector(CoreAccountSelectors.isBusinessAccount, true);

  // act
  const result$ = effects.loadActivities$;

  // assert
  result$.subscribe((response) => {
    expect(response).toMatchSnapshot();
    expect(activityStreamSpy).toHaveBeenNthCalledWith(
      1,
      ActivityArea.STREAM,
      0,
      DEFAULT_PAGE_SIZE
    );
  });
}));
```

```ts
exports[`ActivitiesEffects loadActivities should return activities 1`] = `
Object {
  "activities": Array [
    Object {
      "additionalText": undefined,
      "avatarUrl": undefined,
      "createdAt": undefined,
      "forBusinessAccount": false,
      "iconName": undefined,
      "id": "id1",
      "imageUrl": undefined,
      "message": "translation-template",
      "placeholderValues": Object {},
      "previewImageUrl": undefined,
      "recordId": undefined,
      "title": "",
    },
  ],
  "type": "[Activities] Load Activities Remotely Success",
}
`;
```

THE EASIER IT IS TO TEST THE BETTER THE CODEBASE WILL BE TESTED

Q & A

THANK YOU

MARCEL VON WYL

FRONTEND TECH LEAD

MARCEL.VONWYL@CSS.CH