

NVIDIA Omniverse 입문 및 Simulation 기초 실습

신 동 훈

Prof. and Director

Research in Intelligent Mobility Systems Lab.

Korea Maritime & Ocean University

© copyrights KMOU RiMS

2025. 02. 28

(Proudly Assisted by 윤택한, 우태걸)

Contents

1. **NVIDIA Omniverse 활용, 실습 주제 소개 및 응용 예시**
2. **Omniverse 내 데이터 추출 및 적용 방법 소개**
3. **Action Graph 기반 데이터 추출 로직 구현 및 실습**
 - Ground Plane 및 Vehicle 모델 생성
 - Vehicle Velocity 데이터 추출을 위한 Action Graph
 - Vehicle Prim의 Velocity 데이터 가져오기
 - 불러온 차량 Velocity 데이터 String 타입으로 변환 및 결합
 - Python Script를 통한 Vehicle Velocity Data 실시간 추출
4. **Action Graph 기반 데이터 적용 로직 구현**
 - 추출된 데이터 적용을 위한 Action Graph
 - Python Script를 통한 추출된 Vehicle Velocity Data Import
 - Import된 Vehicle Velocity 데이터를 Vehicle Prim에 적용
5. **추출된 데이터 기반 Simulation 실습**
6. **Q & A**

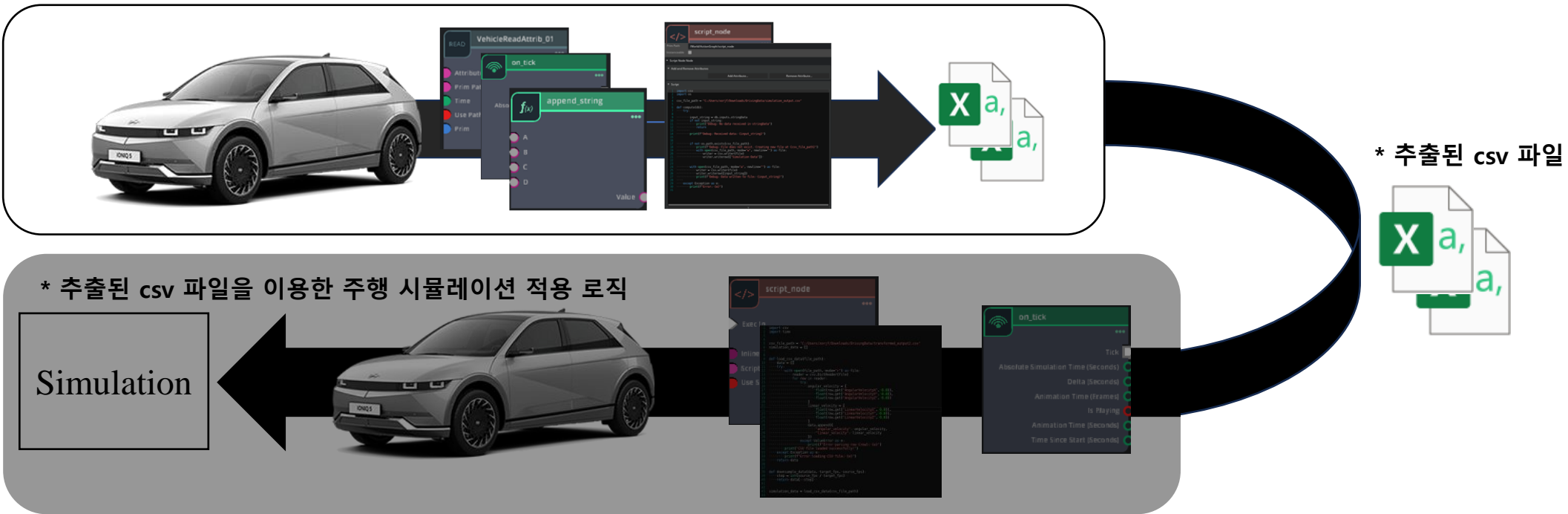
Contents

1. NVIDIA Omniverse 활용, 실습 주제 소개 및 응용 예시
2. Omniverse 내 데이터 추출 및 적용 방법 소개
3. Action Graph 기반 데이터 추출 로직 구현 및 실습
 - Ground Plane 및 Vehicle 모델 생성
 - Vehicle Velocity 데이터 추출을 위한 Action Graph
 - Vehicle Prim의 Velocity 데이터 가져오기
 - 불러온 차량 Velocity 데이터 String 타입으로 변환 및 결합
 - Python Script를 통한 Vehicle Velocity Data 실시간 추출
4. Action Graph 기반 데이터 적용 로직 구현
 - 추출된 데이터 적용을 위한 Action Graph
 - Python Script를 통한 추출된 Vehicle Velocity Data Import
 - Import된 Vehicle Velocity 데이터를 Vehicle Prim에 적용
5. 추출된 데이터 기반 Simulation 실습
6. Q & A

NVIDIA Omniverse 내 기능을 활용한 데이터 추출/적용 방법 소개

- Isaac sim Physics Vehicle을 활용한 차량 데이터 추출 및 데이터 적용 방법

* 주행데이터 csv 파일 추출 로직



- Omniverse 시뮬레이션 차량 주행 데이터 추출을 위한 로직 구현
 - 실제 주행 시뮬레이션 내에서 차량의 속도, 가속도, 각속도 등의 파라미터 값들을 실시간으로 확인 및 추출할 수 있는 기반 구축
 - 시뮬레이션 차량과 Omniverse 시뮬레이션 내 기능인 Action Graph와 Python Scripts의 상호작용을 통해 주행 데이터 추출

NVIDIA Omniverse 내 기능을 활용한 데이터 추출/적용 방법 소개

- Isaac sim Physics Vehicle을 활용한 차량 데이터 추출 및 데이터 적용 방법

* 주행데이터 csv 파일 추출 로직



* 추출된 csv 파일



* 추출된 csv 파일을 이용한 주행 시뮬레이션 적용 로직



- Omniverse 시뮬레이션 차량 주행 데이터 적용을 위한 로직 구현

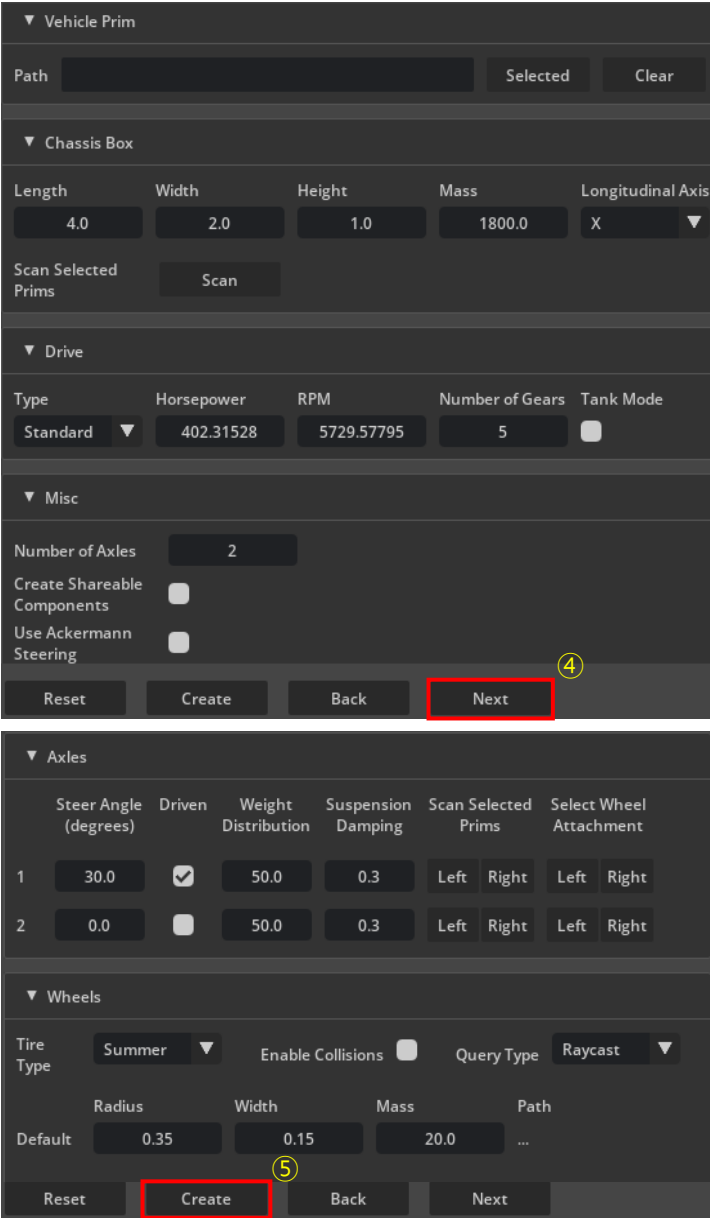
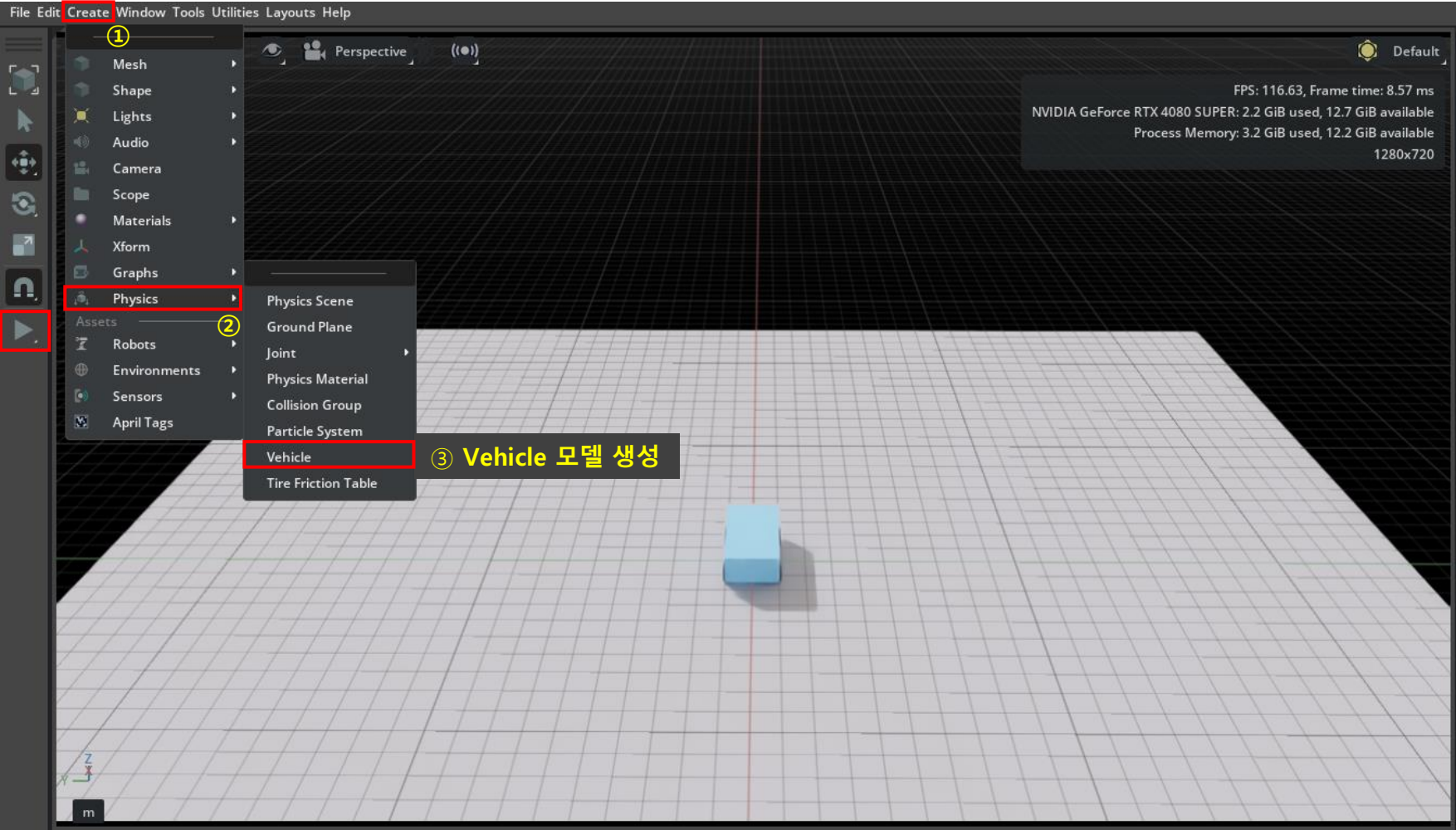
- 실제 주행 시뮬레이션 내에서 차량의 속도, 가속도, 각속도 등의 파라미터 값들을 실시간으로 적용할 수 있는 기반 구축
- 시뮬레이션 차량에 추출된 데이터를 Action Graph와 Python Scripts의 상호작용을 통해 가상 주행 시뮬레이션 실행

Contents

1. NVIDIA Omniverse 활용, 실습 주제 소개 및 응용 예시
2. Omniverse 내 데이터 추출 및 적용 방법 소개
3. **Action Graph 기반 데이터 추출 로직 구현 및 실습**
 - Ground Plane 및 Vehicle 모델 생성
 - Vehicle Velocity 데이터 추출을 위한 Action Graph
 - Vehicle Prim의 Velocity 데이터 가져오기
 - 불러온 차량 Velocity 데이터 String 타입으로 변환 및 결합
 - Python Script를 통한 Vehicle Velocity Data 실시간 추출
4. Action Graph 기반 데이터 적용 로직 구현
 - 추출된 데이터 적용을 위한 Action Graph
 - Python Script를 통한 추출된 Vehicle Velocity Data Import
 - Import된 Vehicle Velocity 데이터를 Vehicle Prim에 적용
5. 추출된 데이터 기반 Simulation 실습
6. Q & A

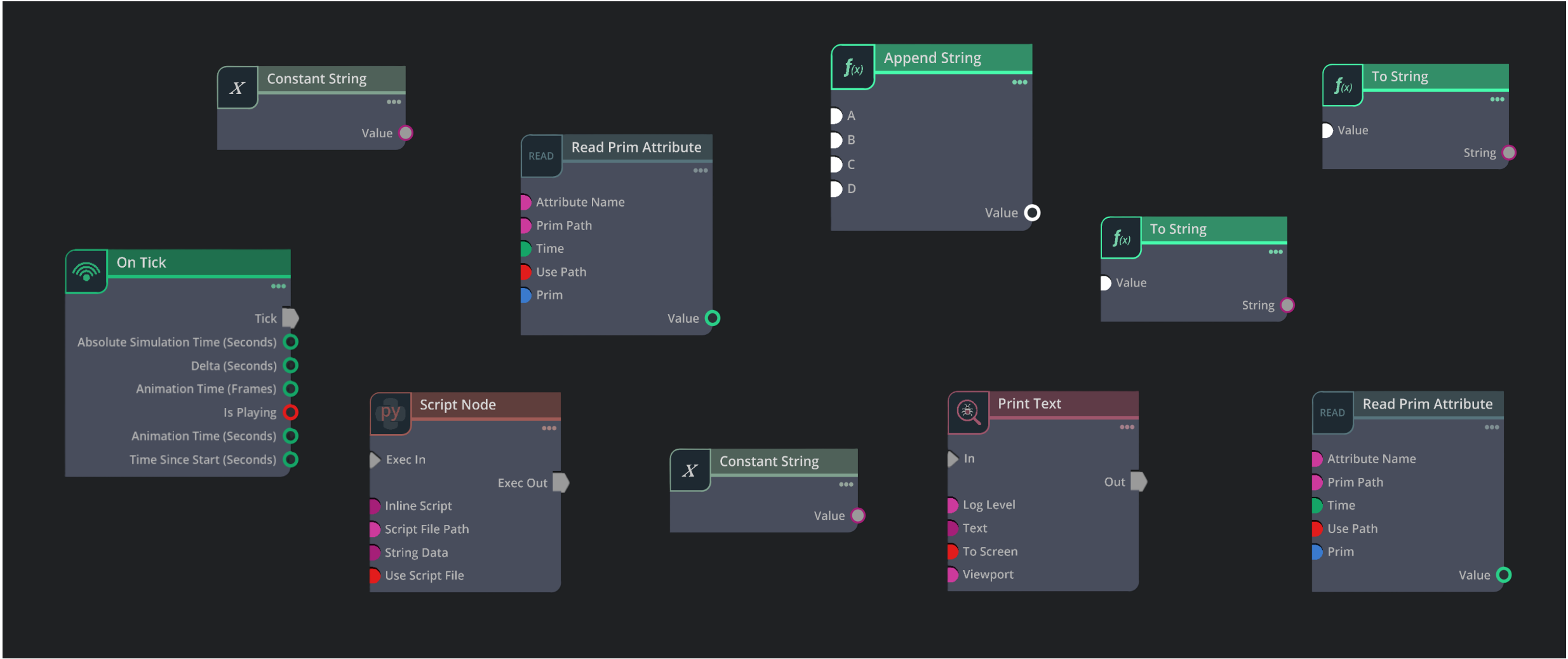
Action Graph 기반 데이터 추출 로직 구현 및 실습

- Vehicle 모델 생성



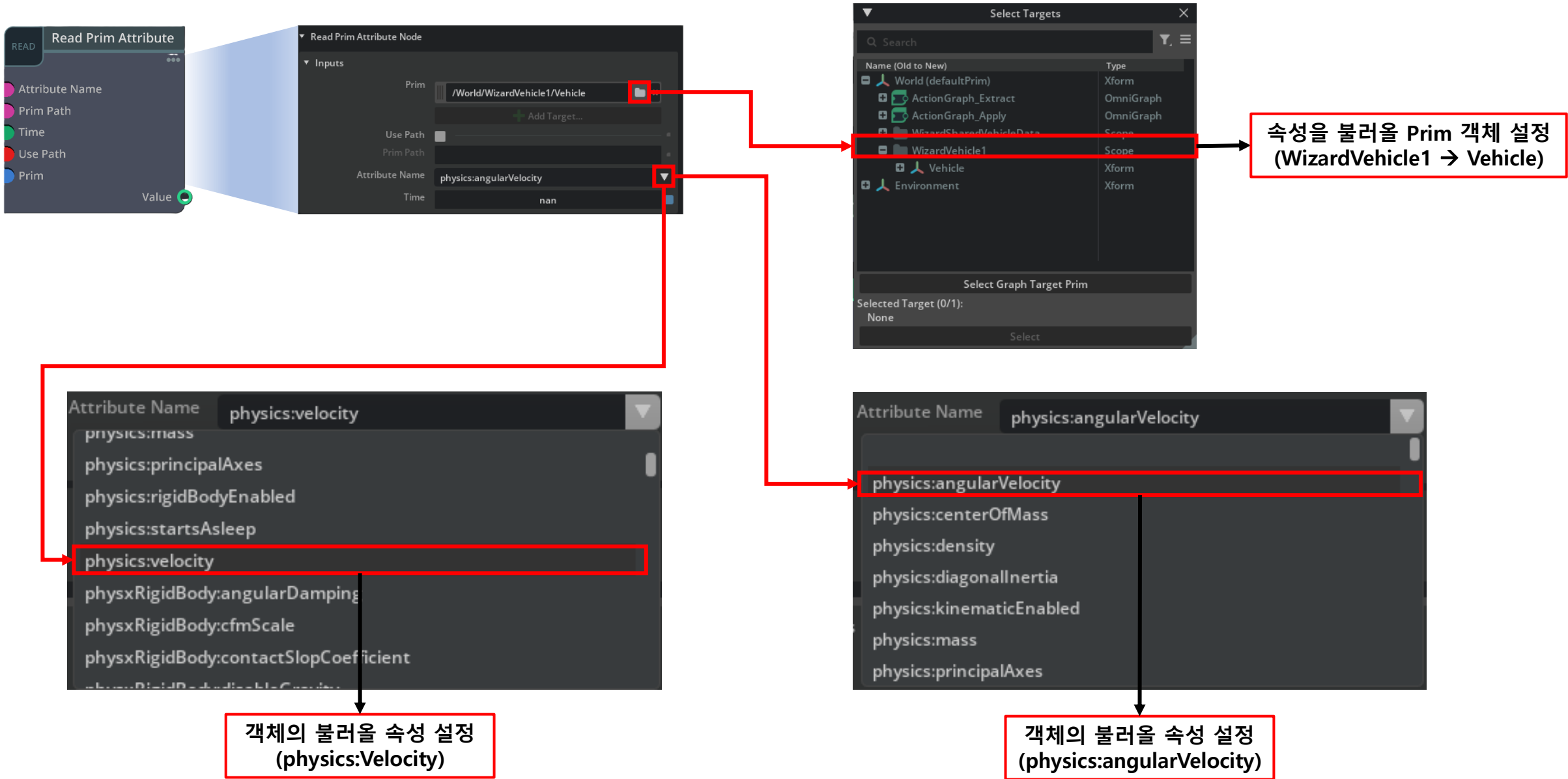
Action Graph 기반 데이터 추출 로직 구현 및 실습: Quiz

- Vehicle Velocity 데이터 추출을 위한 Action Graph



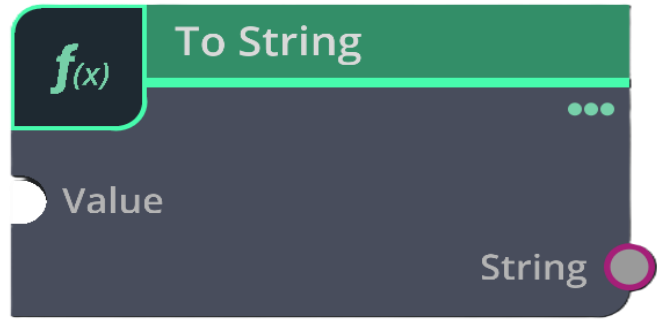
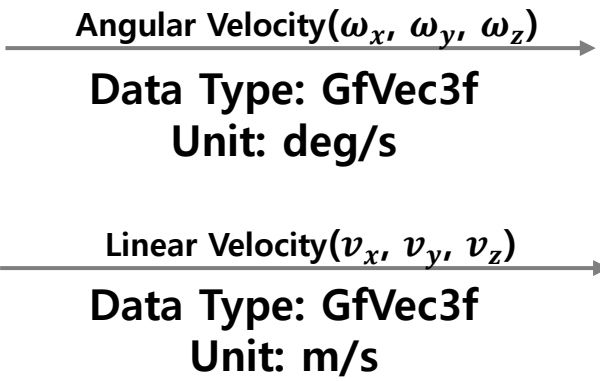
Action Graph 기반 데이터 추출 로직 구현 및 실습: 데이터 추출 노드 [1/3]

- Vehicle Prim의 Linear/Angular Velocity 데이터 추출 노드



Action Graph 기반 데이터 추출 로직 구현 및 실습: 데이터 추출 노드 [2/3]

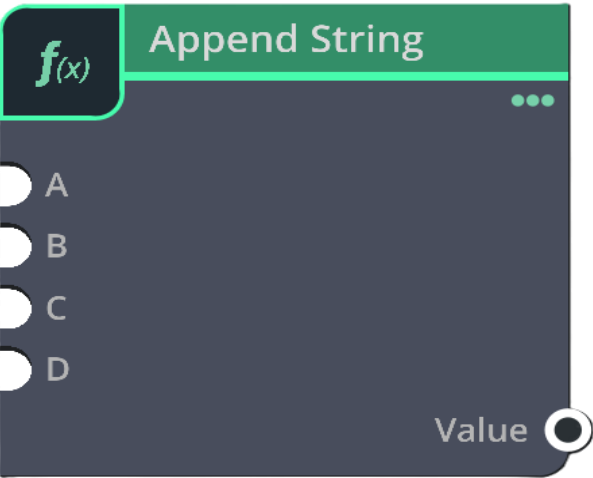
- 추출된 Linear/Angular Velocity 데이터의 String 타입 변환 및 결합 노드



<추출된 데이터를 String 타입으로 강제 형변환 시키는 노드 >



<시뮬레이션 화면 내 Default로 출력되는 String 설정 노드>



<여러 String 타입의 데이터를 하나로 결합 시켜주는 노드>

실시간 추출된 차량 데이터 (Data Type: String)

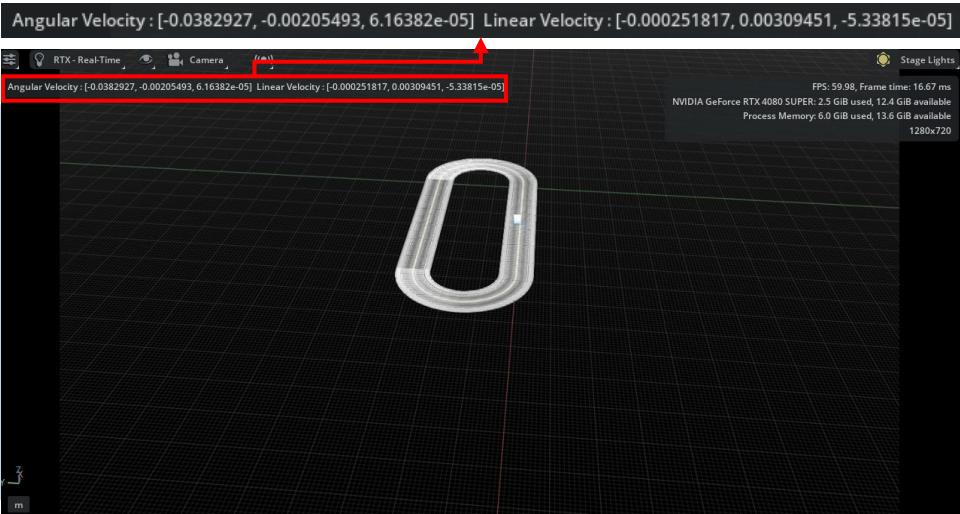
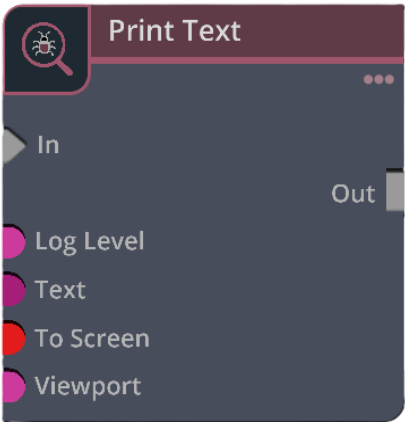
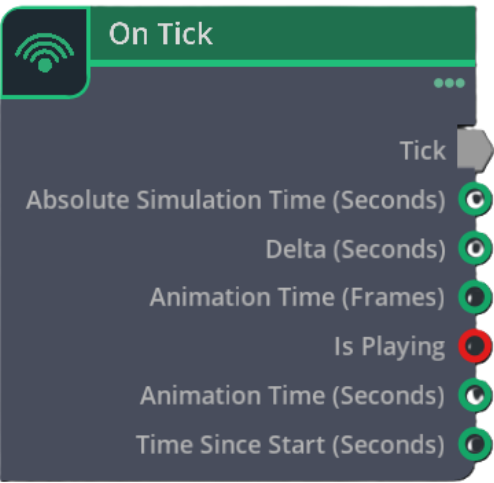
Angular Velocity: [$\omega_x, \omega_y, \omega_z$] Linear Velocity: [v_x, v_y, v_z]

Angular Velocity : [0.0157618, 0.741693, 0.00769744] Linear Velocity : [2.4416, -0.00873407, -0.0118046]

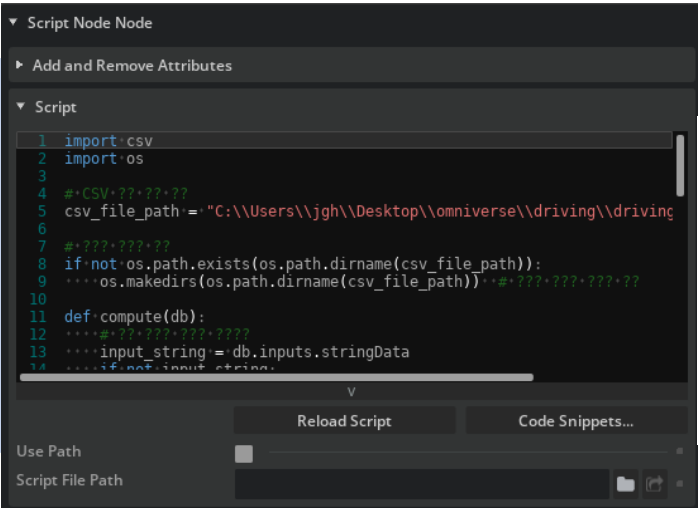
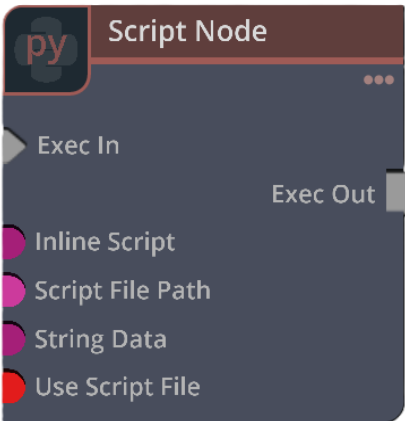
추후, 시뮬레이션 실행결과 위 이미지와 같이 화면 상단에 String 형태로 결합된 주행 데이터가 실시간 출력되면 성공!

Action Graph 기반 데이터 추출 로직 구현 및 실습: 데이터 추출 노드 [3/3]

- Python Script를 통한 Vehicle Velocity Data 실시간 추출



실시간 Velocity Data Monitoring 가능



csv file

Python Script를 통한 Vehicle Velocity Data 추출 가능

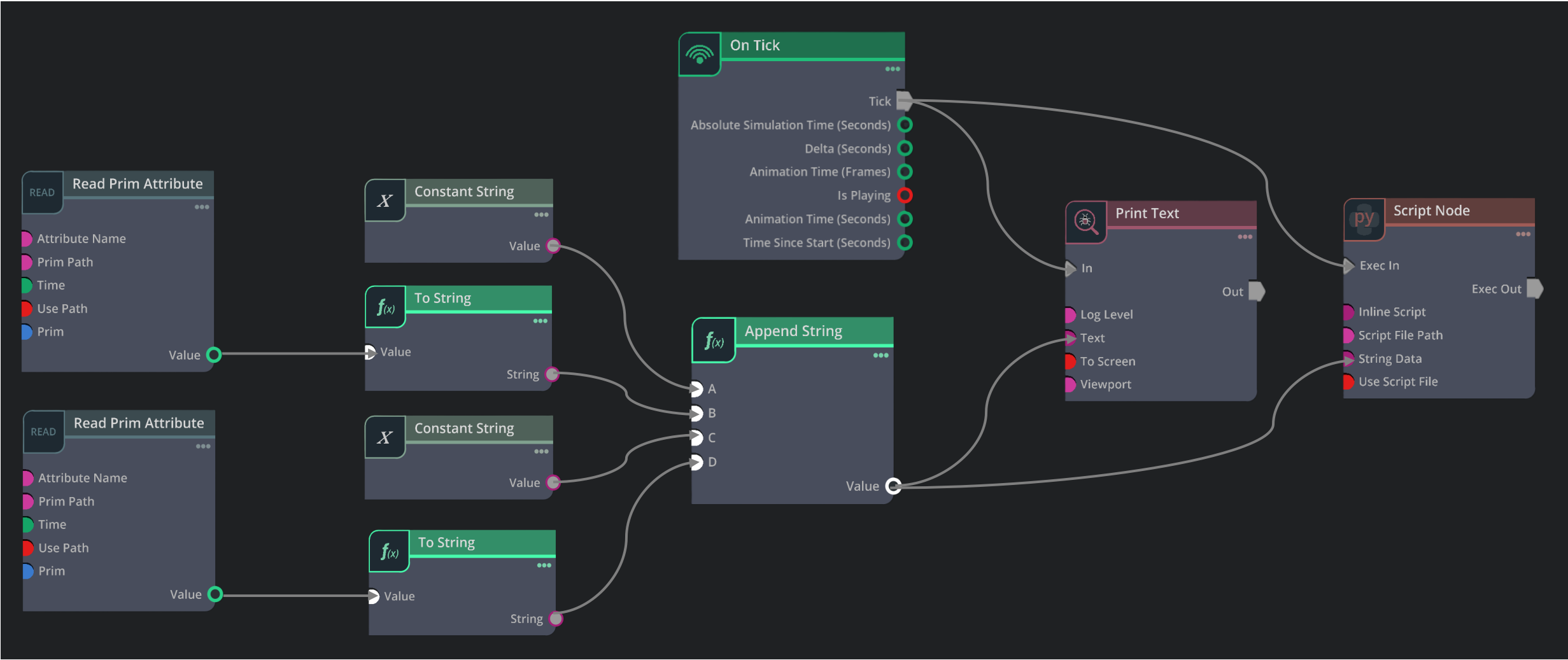
<On Tick>

: 시뮬레이션이 실행되는 동안, 매 프레임마다
노드를 실행시키는 트리거 역할 수행

실시간 실행이 요구되는 모든 노드에게
가장 중요한 노드!!

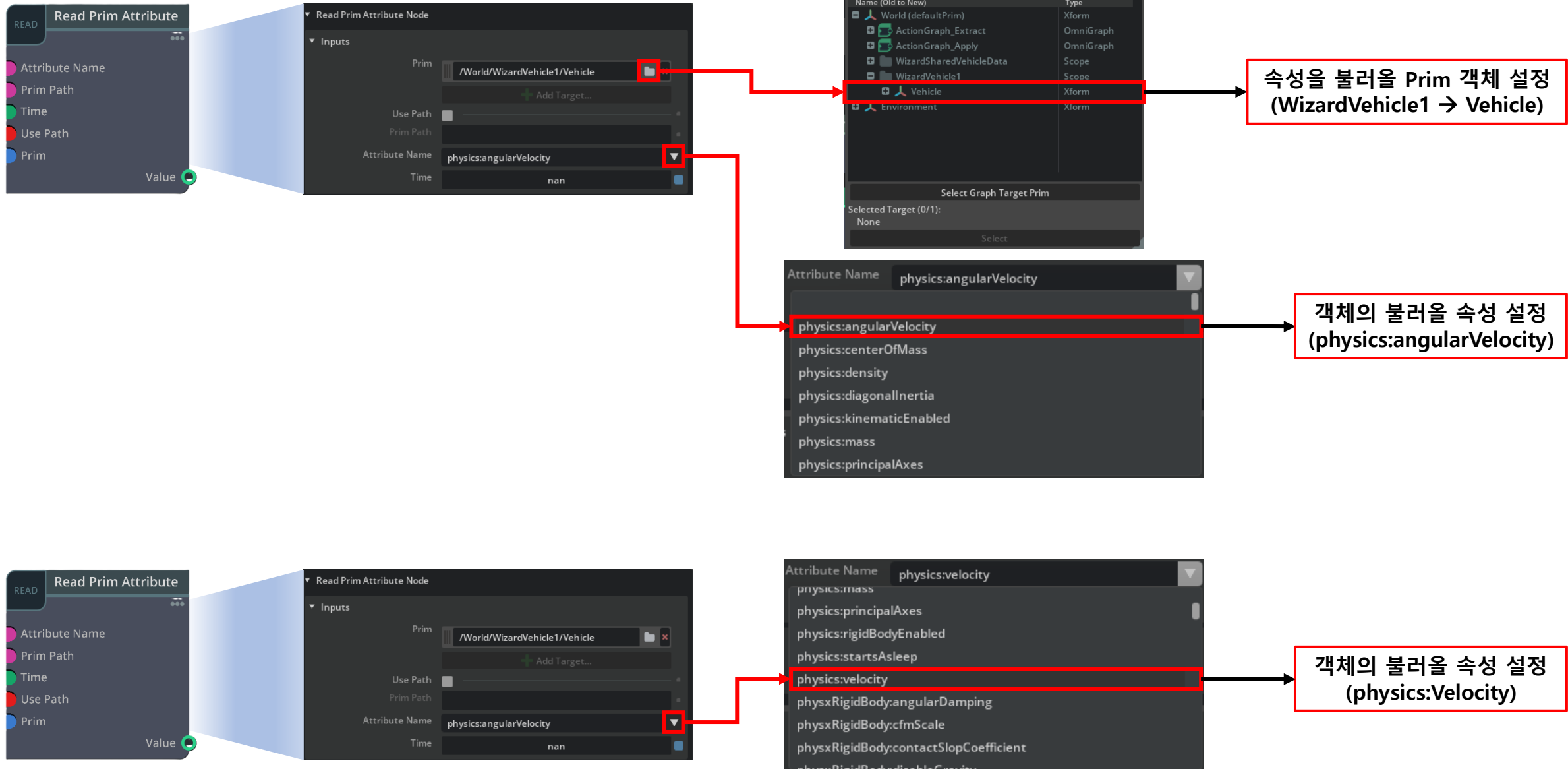
Action Graph 기반 데이터 추출 로직 구현 및 실습: Answer

- Vehicle Velocity 데이터 추출을 위한 Action Graph



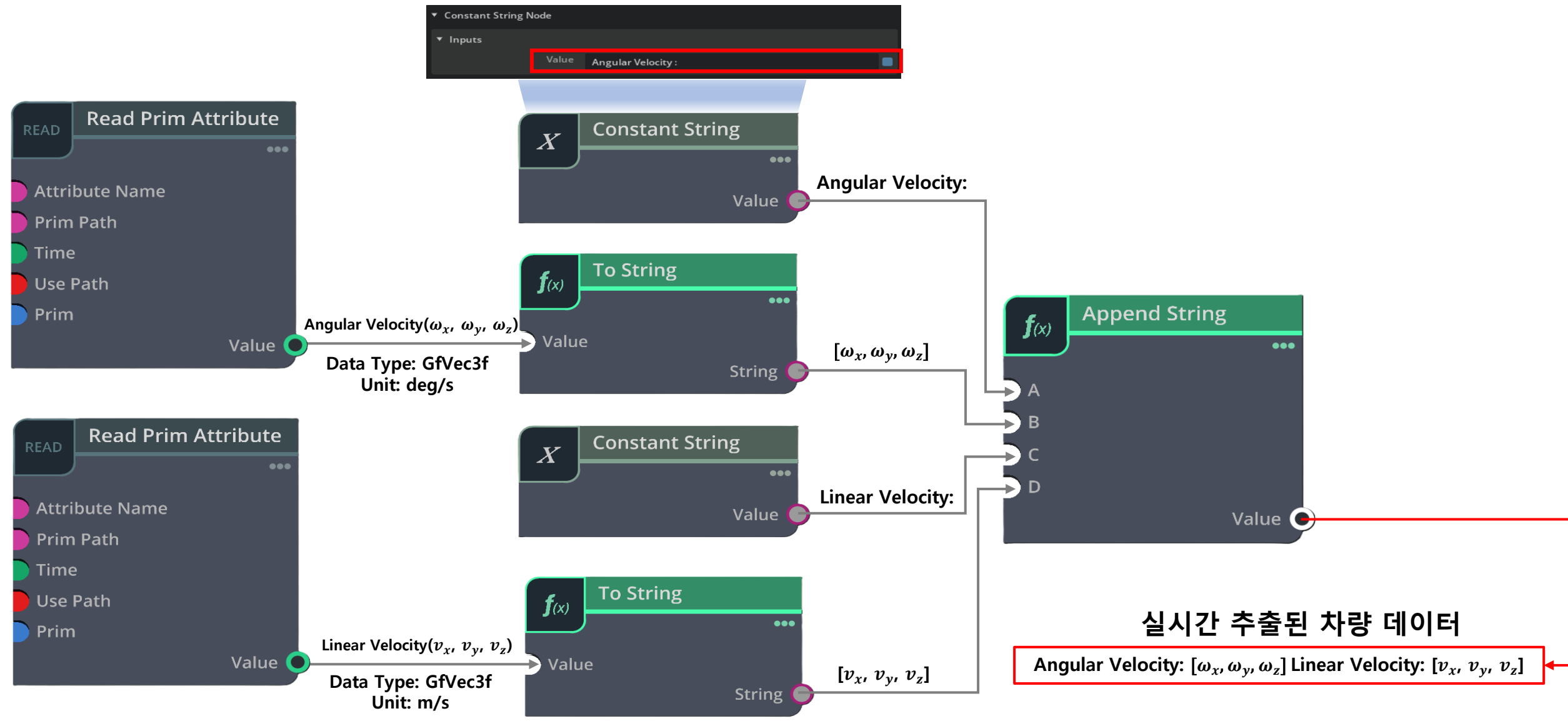
Action Graph 기반 데이터 추출 로직 구현 및 실습 [1/3]

- Vehicle Prim의 Velocity 데이터 가져오기



Action Graph 기반 데이터 추출 로직 구현 및 실습 [2/3]

- 불러온 차량 Velocity 데이터 String 타입으로 변환 및 결합



Action Graph 기반 데이터 추출 로직 구현 및 실습 [3/3]

- Python Script를 통한 Vehicle Velocity Data 실시간 추출 [1/2]

On Tick

Tick

Absolute Simulation Time (Seconds)

Delta (Seconds)

Animation Time (Frames)

Is Playing

Animation Time (Seconds)

Time Since Start (Seconds)

Append String

A

B

C

D

Value

Print Text

In

Log Level

Text

To Screen

Viewport

Out

Script Node

Exec In

Inline Script

Script File Path

String Data

Use Script File

Exec Out

Angular Velocity : [-0.0382927, -0.00205493, 6.16382e-05] Linear Velocity : [-0.000251817, 0.00309451, -5.33815e-05]

RTX-Real-Time

Camera

Path

Angular Velocity : [-0.0382927, -0.00205493, 6.16382e-05] Linear Velocity : [-0.000251817, 0.00309451, -5.33815e-05]

FPS: 59.98, Frame time: 16.67 ms

NVIDIA GeForce RTX 4080 SUPER: 2.5 GiB used, 12.4 GiB available

Process Memory: 6.0 GiB used, 13.6 GiB available

1280x720

실시간 Velocity Data Monitoring 가능

Script Node Node

Add and Remove Attributes

Script

1 import csv

2 import os

3

4 #CSV:??*??*??

5 csv_file_path = "C:\\Users\\jgh\\Desktop\\omniverse\\driving\\driving

6

7 #??*??*??

8 if not os.path.exists(os.path.dirname(csv_file_path)):

9 os.makedirs(os.path.dirname(csv_file_path))

10

11 def compute(db):

12 #??*??*??*??

13 input_string = db.inputs.stringData

14 if not input_string:

15

16

17

18

19

20

21

22

23

24

25

26

27

28

29

30

31

32

33

34

35

36

37

38

39

40

41

42

43

44

45

46

47

48

49

50

51

52

53

54

55

56

57

58

59

60

61

62

63

64

65

66

67

68

69

70

71

72

73

74

75

76

77

78

79

80

81

82

83

84

85

86

87

88

89

90

91

92

93

94

95

96

97

98

99

100

101

102

103

104

105

106

107

108

109

110

111

112

113

114

115

116

117

118

119

120

121

122

123

124

125

126

127

128

129

130

131

132

133

134

135

136

137

138

139

140

141

142

143

144

145

146

147

148

149

150

151

152

153

154

155

156

157

158

159

160

161

162

163

164

165

166

167

168

169

170

171

172

173

174

175

176

177

178

179

180

181

182

183

184

185

186

187

188

189

190

191

192

193

194

195

196

197

198

199

200

201

202

203

204

205

206

207

208

209

210

211

212

213

214

215

216

217

218

219

220

221

222

223

224

225

226

227

228

229

230

231

232

233

234

235

236

237

238

239

240

241

242

243

244

245

246

247

248

249

250

251

252

253

254

255

256

257

258

259

260

261

262

263

264

265

266

267

268

269

270

271

272

273

274

275

276

277

278

279

280

281

282

283

284

285

286

287

288

289

290

291

292

293

294

295

296

297

298

299

300

301

302

303

304

305

306

307

308

309

310

311

312

313

314

315

316

317

318

319

320

321

322

323

324

325

326

327

328

329

330

331

332

333

334

335

336

337

338

339

340

341

342

343

344

345

346

347

348

349

350

351

352

353

354

355

356

357

358

359

360

361

362

363

364

365

366

367

368

369

370

371

372

373

374

375

376

377

378

379

380

381

382

383

384

385

386

387

388

389

390

391

392

393

394

395

396

397

398

399

400

401

402

403

404

405

406

407

408

409

410

411

412

413

414

415

416

417

418

419

420

421

422

423

424

425

426

427

428

429

430

431

432

433

434

435

436

437

438

439

440

441

442

443

444

445

446

447

448

449

450

451

452

453

454

455

456

457

458

459

460

461

462

463

464

465

466

467

468

469

470

471

472

473

474

475

476

477

478

479

480

481

482

483

484

485

486

487

488

489

490

491

492

493

494

495

496

497

498

499

500

501

502

503

504

505

506

507

508

509

510

511

512

513

514

515

516

517

518

519

520

521

522

523

524

525

526

527

528

529

530

531

532

533

534

535

536

537

538

539

540

541

542

543

544

545

546

547

548

549

550

551

552

553

554

555

556

557

558

559

560

561

562

563

564

565

566

567

568

569

570

571

572

573

574

575

576

577

578

579

580

581

582

583

584

585

586

587

588

589

590

591

592

593

594

595

596

597

598

599

600

601

602

603

604

605

606

607

608

609

610

611

612

613

614

615

616

617

618

619

620

621

622

623

624

625

626

627

628

629

630

631

632

633

634

635

636

637

638

639

640

641

642

643

644

645

646

647

648

649

650

651

652

653

654

655

656

657

658

659

660

661

662

663

664

665

666

667

668

669

670

671

672

673

674

675

676

677

678

679

680

681

682

683

684

685

686

687

688

689

690

691

692

693

694

695

696

697

698

699

700

701

702

703

704

705

706

707

708

709

710

711

712

713

714

715

716

717

718

719

720

721

722

723

724

725

726

727

728

729

730

731

732

733

734

735

736

737

738

739

740

741

742

743

744

745

746

747

748

749

750

751

752

753

754

755

756

757

758

759

760

761

762

763

764

765

766

767

768

769

770

771

772

773

774

775

776

777

778

779

780

781

782

783

784

785

786

787

788

789

790

791

792

793

794

795

796

797

798

799

800

801

802

803

804

805

806

807

808

809

810

811

812

813

814

815

816

817

818

819

820

821

822

823

824

825

826

827

828

829

830

831

832

833

834

835

836

837

838

839

840

841

842

843

844

845

846

847

848

849

850

851

852

853

854

855

856

857

858

859

860

861

862

863

864

865

866

867

868

869

870

871

872

873

874

875

876

877

878

879

880

881

882

883

884

885

886

887

888

889

890

891

892

893

894

895

896

897

898

899

900

901

902

903

904

905

906

907

908

909

910

911

912

913

914

915

916

917

918

919

920

921

922

923

924

925

926

927

928

929

930

931

932

933

934

935

936

937

938

939

940

941

942

943

944

945

946

947

948

949

950

951

952

953

954

955

956

957

958

959

960

961

962

963

964

965

966

967

968

969

970

971

972

973

974

975

976

977

978

979

980

981

982

983

984

985

Use Path

Script File Path

Xa,

csv file

Python Script를 통한 Vehicle Velocity Data 추출 가능

Action Graph 기반 데이터 추출 로직 구현 및 실습 [3/3]

• Python Script를 통한 Vehicle Velocity Data 실시간 추출 [2/2]

```
import csv
import os

# CSV 파일 경로 (사용자 환경에 맞게 변경 필요)
#예시: csv_file_path =
"C:\\Users\\jgh\\Desktop\\omniverse\\driving\\driving_data_velocity.csv"
csv_file_path = "C:\\path\\to\\your\\csvfile.csv"

# 폴더가 없으면 생성
if not os.path.exists(os.path.dirname(csv_file_path)):
    os.makedirs(os.path.dirname(csv_file_path)) # 지정된 경로의 폴더를 생성

def compute(db):
    # 입력 문자열 데이터 가져오기
    input_string = db.inputs.stringData
    if not input_string:
        print("Debug: No data received in stringData") # 데이터가 없을 경우 경고 출력
        return

    # 수신된 데이터 출력
    print(f"Debug: Received data: {input_string}")

    # 입력 문자열에서 Angular Velocity와 Linear Velocity 추출
    try:
        # 문자열을 파싱하여 Angular Velocity 부분 추출
        angular_velocity_part = input_string.split("Angular Velocity : ")[1].split(" Linear Velocity : ")[0]
        # Linear Velocity 부분 추출
        linear_velocity_part = input_string.split("Linear Velocity : ")[1]
```

```
# 문자열을 리스트 형태로 변환
angular_velocity = eval(angular_velocity_part)
linear_velocity = eval(linear_velocity_part)

# CSV 파일이 존재하지 않으면 헤더 작성
if not os.path.exists(csv_file_path):
    print(f"Debug: File does not exist. Creating new file at {csv_file_path}")
    with open(csv_file_path, mode="w", newline="") as file:
        writer = csv.writer(file)
        # CSV 헤더 작성
        writer.writerow(["Angular Velocity X", "Angular Velocity Y", "Angular Velocity Z", "Linear Velocity X", "Linear Velocity Y", "Linear Velocity Z"]) # 헤더 추가

# 데이터 저장
with open(csv_file_path, mode="a", newline="") as file:
    writer = csv.writer(file)
    # Angular Velocity와 Linear Velocity 값을 CSV 파일에 추가
    writer.writerow([angular_velocity[0], angular_velocity[1], angular_velocity[2], linear_velocity[0], linear_velocity[1], linear_velocity[2]]) # 데이터 추가
    print(f"Debug: Data written to file: {angular_velocity}, {linear_velocity}") # 저장된 데이터 출력

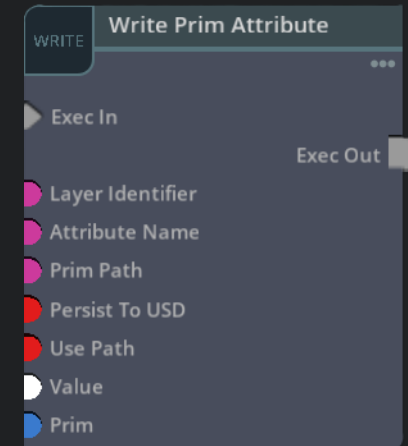
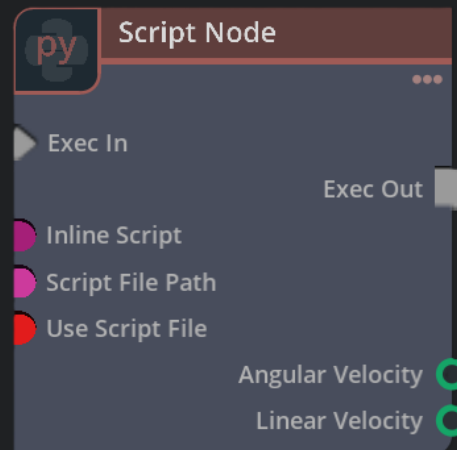
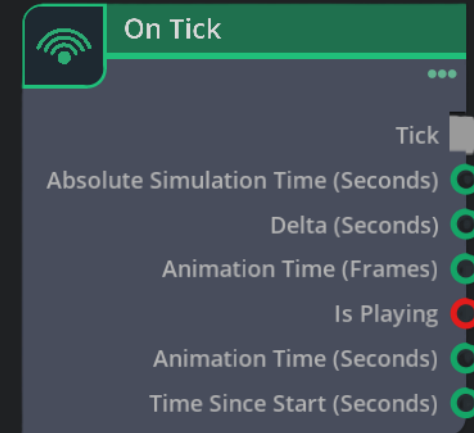
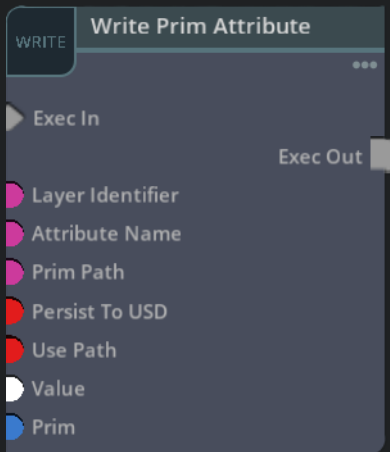
except Exception as e:
    print(f"Error: {e}") # 오류 발생 시 오류 메시지 출력
```


Contents

1. NVIDIA Omniverse 활용, 실습 주제 소개 및 응용 예시
2. Omniverse 내 데이터 추출 및 적용 방법 소개
3. Action Graph 기반 데이터 추출 로직 구현 및 실습
 - Ground Plane 및 Vehicle 모델 생성
 - Vehicle Velocity 데이터 추출을 위한 Action Graph
 - Vehicle Prim의 Velocity 데이터 가져오기
 - 불러온 차량 Velocity 데이터 String 타입으로 변환 및 결합
 - Python Script를 통한 Vehicle Velocity Data 실시간 추출
4. Action Graph 기반 데이터 적용 로직 구현
 - 추출된 데이터 적용을 위한 Action Graph
 - Python Script를 통한 추출된 Vehicle Velocity Data Import
 - Import된 Vehicle Velocity 데이터를 Vehicle Prim에 적용
5. 추출된 데이터 기반 Simulation 실습
6. Q & A

Action Graph 기반 데이터 적용 로직 구현: Quiz

- 추출된 데이터 적용을 위한 Action Graph



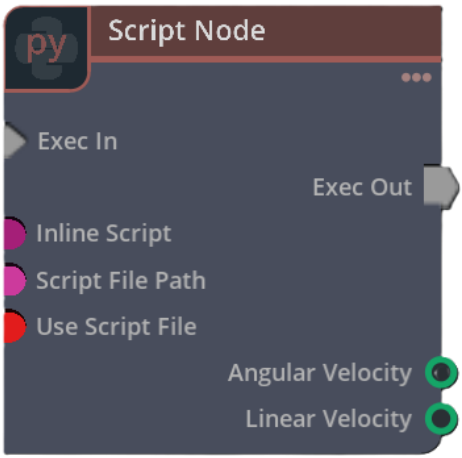
Action Graph 기반 데이터 적용 로직 구현: 데이터 적용 노드 [1/2]

- Python Script를 통한 추출된 Vehicle Velocity Data Import

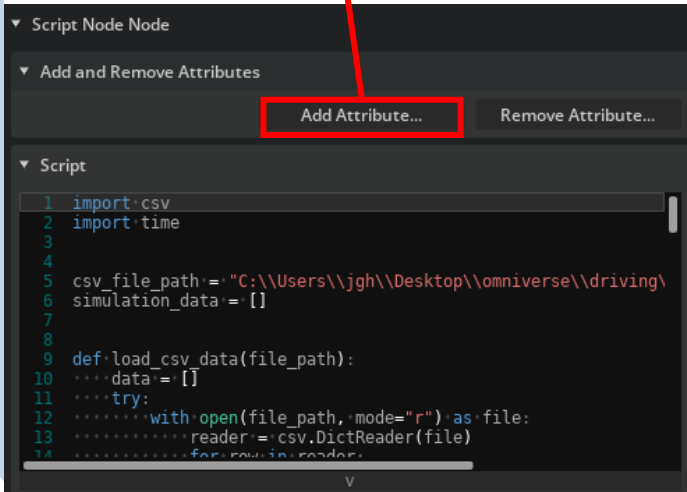
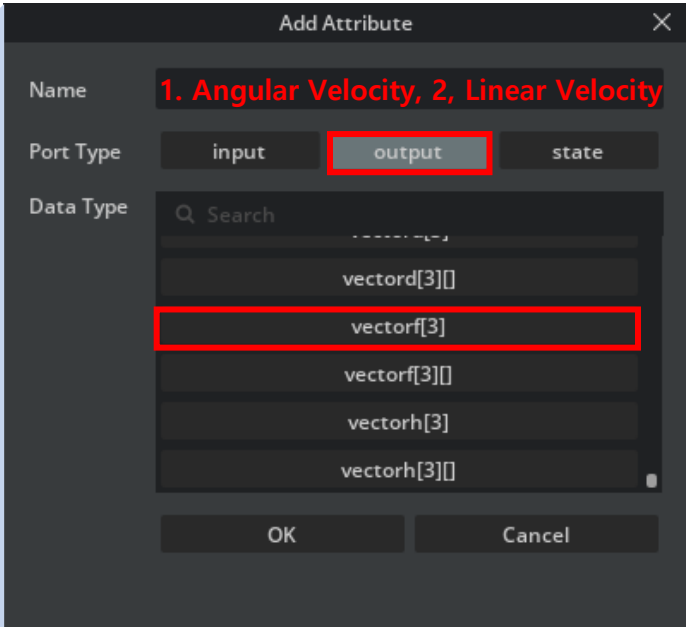
?

→

시뮬레이션이 실행되는 동안,
매 프레임마다 노드를 실행시키는
트리거 역할 수행 노드 연결

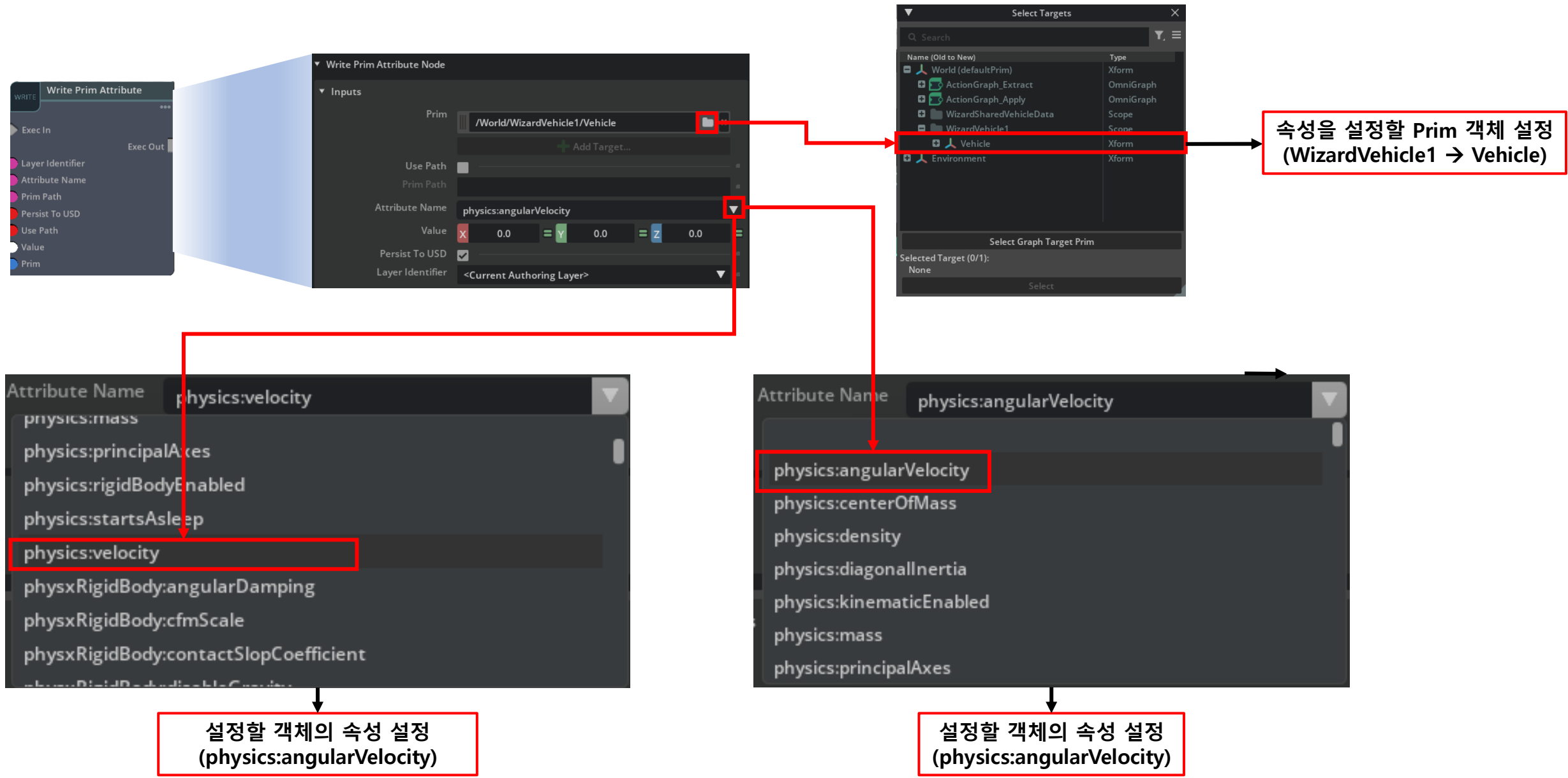


Python Script를 통해,
Angular Velocity, Linear Velocity data를
매 프레임마다 Omniverse 내로 Import



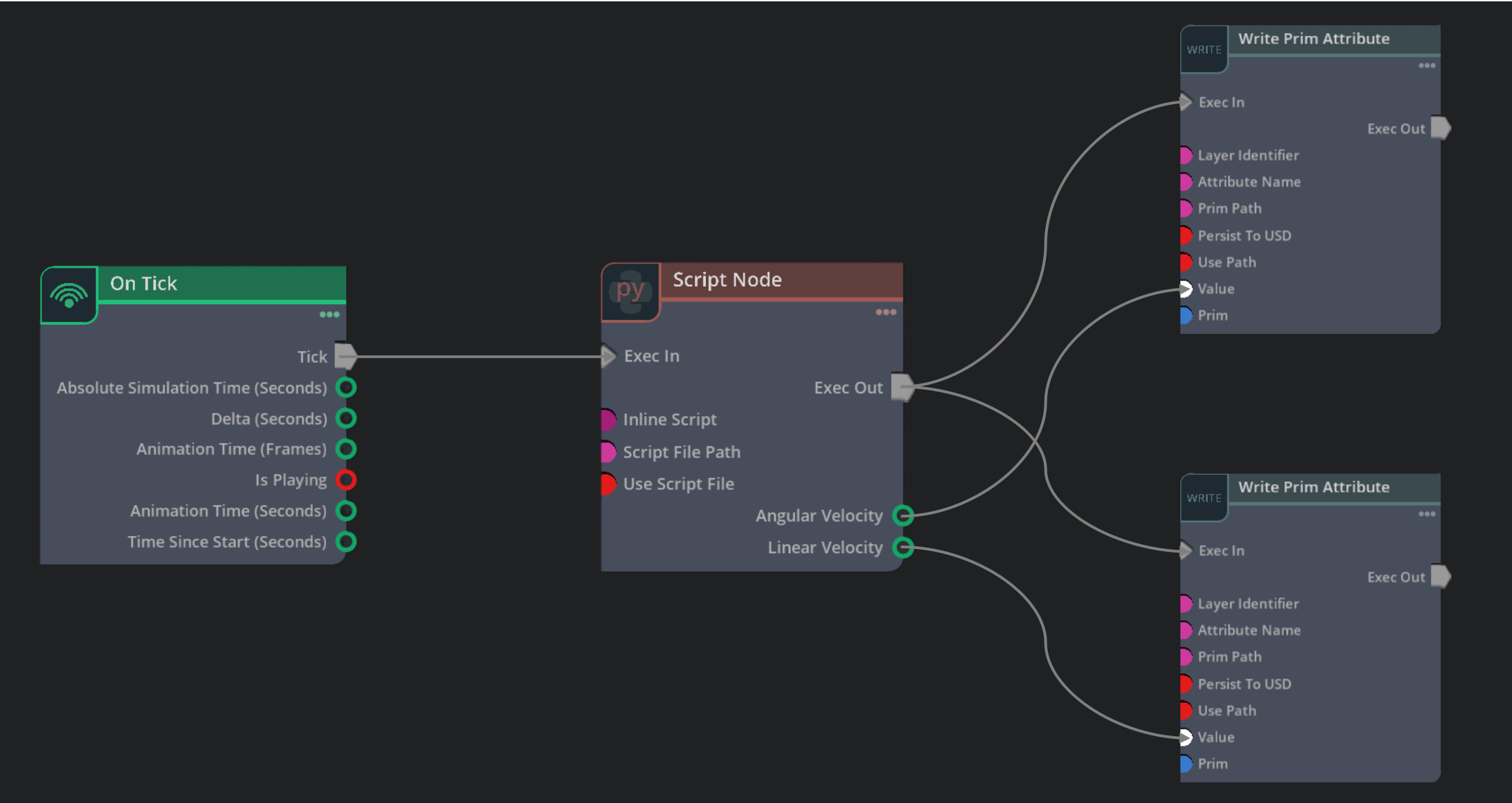
Action Graph 기반 데이터 적용 로직 구현: 데이터 적용 노드 [2/2]

- 실시간 Import된 Linear/Angular Velocity 데이터를 Vehicle Prim에 적용



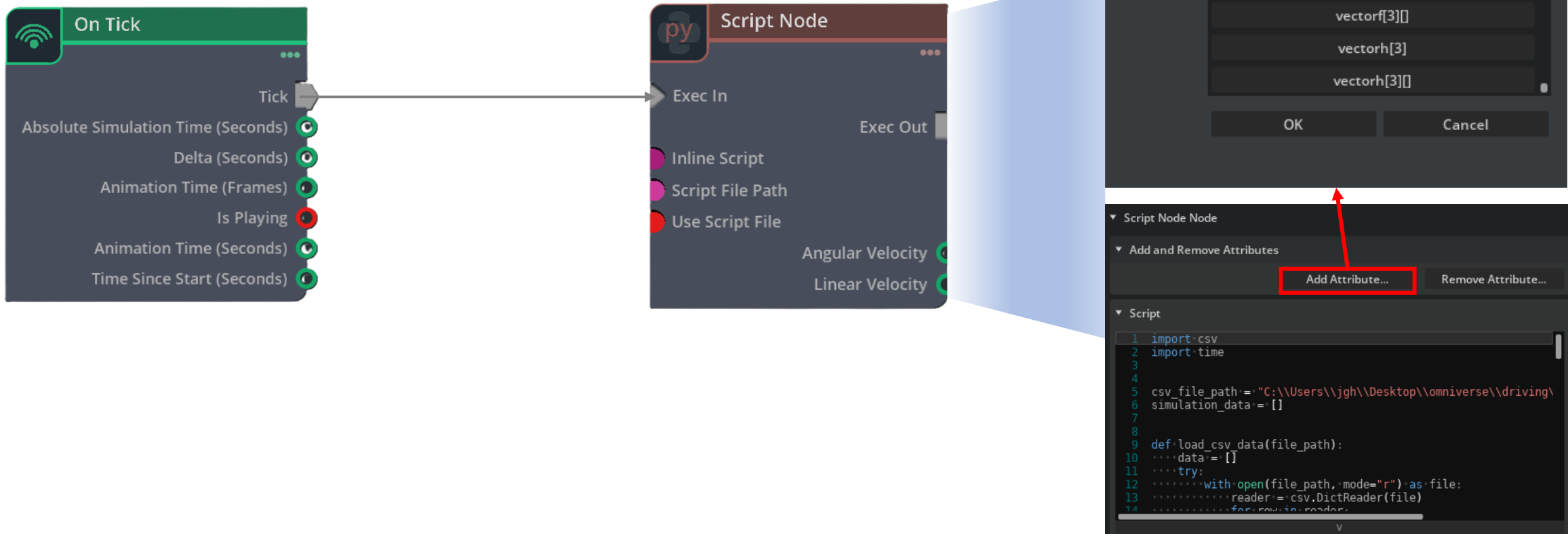
Action Graph 기반 데이터 적용 로직 구현: Answer

- 추출된 데이터 적용을 위한 Action Graph



Action Graph 기반 데이터 적용 로직 구현 [1/3]

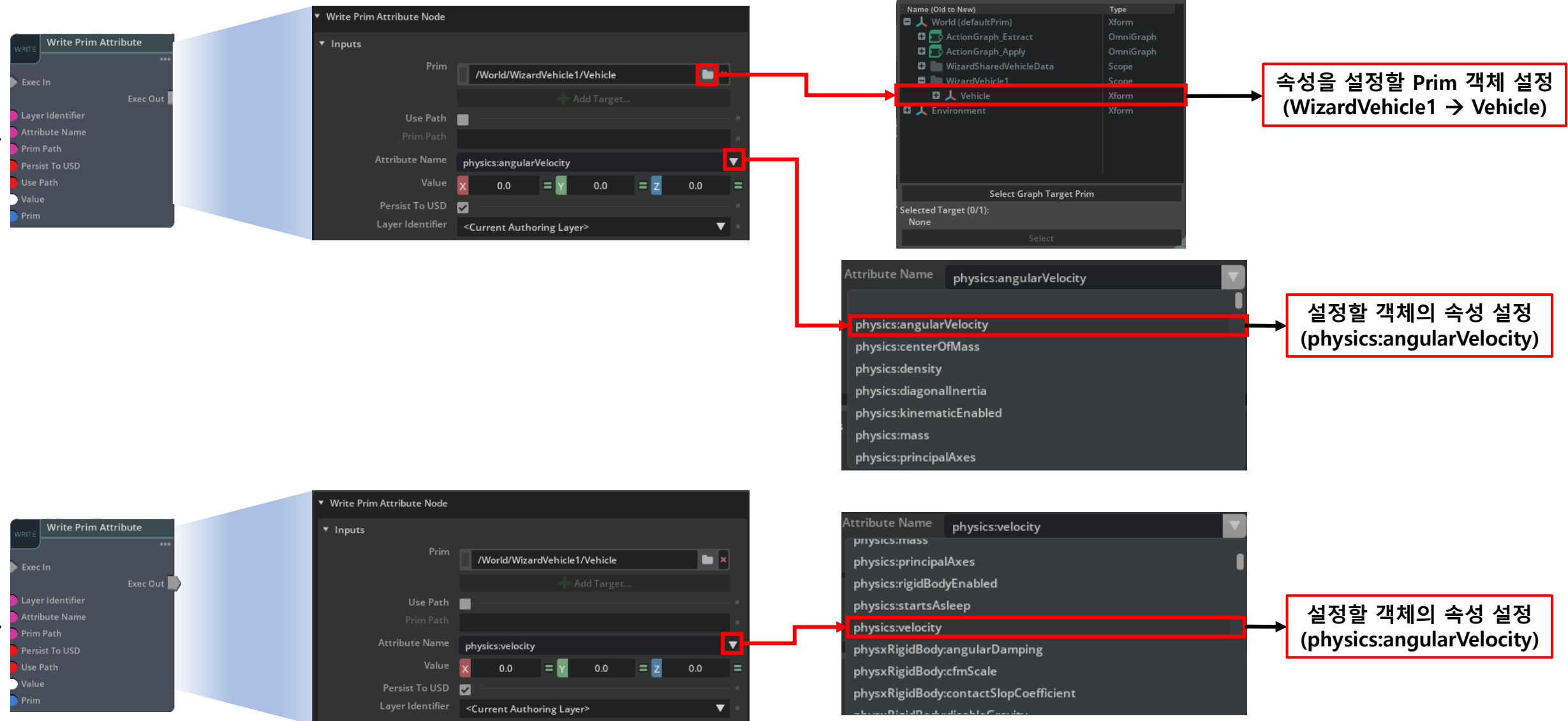
- Python Script를 통한 추출된 Vehicle Velocity Data Import



Python Script를 통한 Angular Velocity, Linear Velocity Import

Action Graph 기반 데이터 적용 로직 구현 [2/3]

- Import된 Vehicle Velocity 데이터를 Vehicle Prim에 적용



Action Graph 기반 데이터 추출 로직 구현 및 실습 [3/3]

- Python Script를 통한 Vehicle Velocity Data 실시간 적용 [1/2]

```
import csv
import time

# CSV 파일 경로 (사용자 환경에 맞게 변경 필요)
csv_file_path = "C:\\path\\to\\your\\csvfile.csv"

# 시뮬레이션 데이터를 저장할 리스트
simulation_data = []

# CSV 데이터 로드
simulation_data = load_csv_data(csv_file_path)

# 원본 및 타겟 프레임 레이트 설정
source_fps = 93.3 # 원본 데이터의 FPS
target_fps = 93.3 # 목표 FPS

# 데이터 다운샘플링 (현재 FPS가 같으므로 변경 없음)
simulation_data = downsample_data(simulation_data, target_fps, source_fps)

# 프레임 지속 시간 계산
target_frame_duration = 1.0 / target_fps # 각 프레임의 시간 간격
last_update_time = time.time() # 마지막 업데이트 시간 기록

frame = 0 # 현재 프레임 인덱스
is_playing = True # 시뮬레이션 실행 여부
```

```
def load_csv_data(file_path):
    """
    CSV 파일을 로드하여 각 행의 데이터를 딕셔너리 리스트로 변환
    """
    data = []
    try:
        with open(file_path, mode="r") as file:
            reader = csv.DictReader(file)
            for row in reader:
                try:
                    # 각 행에서 각 속도 데이터를 가져와서 리스트로 변환
                    angular_velocity = [
                        float(row.get("Angular Velocity X", 0.0)),
                        float(row.get("Angular Velocity Y", 0.0)),
                        float(row.get("Angular Velocity Z", 0.0))]
                    linear_velocity = [
                        float(row.get("Linear Velocity X", 0.0)),
                        float(row.get("Linear Velocity Y", 0.0)),
                        float(row.get("Linear Velocity Z", 0.0))]

                    # 변환된 데이터를 리스트에 추가
                    data.append({
                        "angular_velocity": angular_velocity,
                        "linear_velocity": linear_velocity
                    })
                except ValueError as e:
                    print(f"Error parsing row {row}: {e}")
            print("CSV file loaded successfully!")
    except Exception as e:
        print(f"Error loading CSV file: {e}")
    return data
```


Action Graph 기반 데이터 추출 로직 구현 및 실습 [3/3]

• Python Script를 통한 Vehicle Velocity Data 실시간 적용 [2/2]

```
def downsample_data(data, target_fps, source_fps):  
    """  
    원본 FPS에서 타겟 FPS로 다운샘플링 (프레임 수 줄이기)  
    """  
    step = int(source_fps / target_fps) # 몇 프레임마다 하나를 선택할지 결정  
    return data[::step] # 지정한 간격으로 데이터를 샘플링하여 반환  
  
def compute(db):  
    """  
    프레임마다 실행되는 함수. 시뮬레이션 데이터를 기반으로 출력값을 갱신.  
    """  
    global frame, simulation_data, last_update_time, is_playing  
  
    # 시뮬레이션이 정지된 경우 종료  
    if not is_playing:  
        print("Playback stopped.")  
        return  
  
    # 현재 시간이 마지막 업데이트 시간보다 프레임 간격보다 적다면 대기  
    current_time = time.time()  
    if current_time - last_update_time < target_frame_duration:  
        return # 아직 프레임을 갱신할 시간이 되지 않음  
  
    # 모든 데이터를 재생한 경우 종료  
    if frame >= len(simulation_data):  
        print("End of simulation data. Stopping playback.")  
        db.outputs.angularVelocity = (0.0, 0.0, 0.0) # 정지 상태로 설정  
        db.outputs.linearVelocity = (0.0, 0.0, 0.0)  
        is_playing = False # 재생 중지  
        return
```

```
# 현재 프레임의 데이터 가져오기  
data = simulation_data[frame]  
angular_velocity = data["angular_velocity"]  
linear_velocity = data["linear_velocity"]  
  
# 출력 값 업데이트  
db.outputs.angularVelocity = tuple(angular_velocity)  
db.outputs.linearVelocity = tuple(linear_velocity)  
  
# 10프레임마다 로그 출력  
if frame % 10 == 0:  
    print(f"Frame {frame}: Angular Velocity = {angular_velocity}, Linear  
Velocity = {linear_velocity}")  
  
# 다음 프레임으로 이동  
frame += 1  
last_update_time = current_time
```

Contents

1. NVIDIA Omniverse 활용, 실습 주제 소개 및 응용 예시
2. Omniverse 내 데이터 추출 및 적용 방법 소개
3. Action Graph 기반 데이터 추출 로직 구현 및 실습
 - Ground Plane 및 Vehicle 모델 생성
 - Vehicle Velocity 데이터 추출을 위한 Action Graph
 - Vehicle Prim의 Velocity 데이터 가져오기
 - 불러온 차량 Velocity 데이터 String 타입으로 변환 및 결합
 - Python Script를 통한 Vehicle Velocity Data 실시간 추출
4. Action Graph 기반 데이터 적용 로직 구현
 - 추출된 데이터 적용을 위한 Action Graph
 - Python Script를 통한 추출된 Vehicle Velocity Data Import
 - Import된 Vehicle Velocity 데이터를 Vehicle Prim에 적용
5. 추출된 데이터 기반 Simulation 실습
6. Q & A

추출된 데이터 기반 Simulation 실습

- 추출된 데이터 import를 통한 차량 Simulation

The screenshot displays the Isaac Sim environment. In the center viewport, a white car is positioned on a grey track. The top right corner shows performance metrics: FPS: 116.39, Frame time: 8.59 ms, NVIDIA GeForce RTX 4080 SUPER: 2.4 GiB used, 12.6 GiB available, Process Memory: 2.8 GiB used, 13.6 GiB available, and resolution 1280x720.

On the left sidebar, the 'Simulation' button (a play icon) is highlighted with a red box and labeled ③ Simulation 실행.

At the bottom, the 'Action Graph' editor is open, showing a graph with an 'On Tick' node connected to a 'Script Node'. The 'Script Node' is highlighted with a red box and labeled ① Action Graph 내 Script Node 선택.

On the right sidebar, the 'Script Node Node' editor is open, showing a Python script for loading CSV data. The script is labeled ② Python Script Load. The script content is as follows:

```
1 import csv
2 import time
3
4
5 csv_file_path = "C:\\Users\\jgh\\Desktop\\omniverse\\driving\\driving
6 simulation_data = []
7
8
9 def load_csv_data(file_path):
10     data = []
11     try:
12         with open(file_path, mode="r") as file:
13             reader = csv.DictReader(file)
14             for row in reader:
15                 try:
16                     angular_velocity = [
17                         float(row.get("Angular Velocity X", 0.0)),
18                         float(row.get("Angular Velocity Y", 0.0)),
19                         float(row.get("Angular Velocity Z", 0.0))
20                     ]
21                     linear_velocity = [
22                         float(row.get("Linear Velocity X", 0.0)),
23                         float(row.get("Linear Velocity Y", 0.0)),
24                         float(row.get("Linear Velocity Z", 0.0))
25                     ]
26                     data.append({
27                         "angular_velocity": angular_velocity,
28                         "linear_velocity": linear_velocity
29                     })
30             except ValueError as e:
31                 print(f"Error parsing row {row}: {e}")
32             print("CSV file loaded successfully!")
33     except Exception as e:
34         print(f"Error loading CSV file: {e}")
```

The 'Reload Script' button at the bottom right of the script editor is highlighted with a red box.

Contents

1. NVIDIA Omniverse 활용, 실습 주제 소개 및 응용 예시
2. Omniverse 내 데이터 추출 및 적용 방법 소개
3. Action Graph 기반 데이터 추출 로직 구현 및 실습
 - Ground Plane 및 Vehicle 모델 생성
 - Vehicle Velocity 데이터 추출을 위한 Action Graph
 - Vehicle Prim의 Velocity 데이터 가져오기
 - 불러온 차량 Velocity 데이터 String 타입으로 변환 및 결합
 - Python Script를 통한 Vehicle Velocity Data 실시간 추출
4. Action Graph 기반 데이터 적용 로직 구현
 - 추출된 데이터 적용을 위한 Action Graph
 - Python Script를 통한 추출된 Vehicle Velocity Data Import
 - Import된 Vehicle Velocity 데이터를 Vehicle Prim에 적용
5. 추출된 데이터 기반 Simulation 실습
6. Q & A

Q & A

Thank You