<!DOCTYPE html>

<html lang="ko">

<head>

    <meta charset="UTF-8">

    <meta http-equiv="X-UA-Compatible" content="IE=edge">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Document</title>

    <link rel="preconnect" href="https://fonts.googleapis.com">

    <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>

    <link href="https://fonts.googleapis.com/css2?family=Noto+Sans+KR:wght@400;700&display=swap" rel="stylesheet">

    <link rel="stylesheet" href="./reset.css">

    <link rel="stylesheet" href="./style.css">

</head>

<body id="show-section0">

    <nav class="global-nav">

        <div class="global-nav-links">

            <a href="#" class="global-nav-link">Home</a>

            <a href="#" class="global-nav-link">History</a>

            <a href="#" class="global-nav-link">Store</a>

            <a href="#" class="global-nav-link">Contact</a>

        </div>

    </nav>

    <nav class="local-nav">

        <div class="local-nav-links">

            <a href="#" class="local-nav-product-link">Real Apple Pro</a>

            <a href="#" class="local-nav-link">Overview</a>

            <a href="#" class="local-nav-link">Compare</a>

            <a href="#" class="local-nav-link">Buy</a>

        </div>

    </nav>

    <section class="scroll-section" id="section-0">

        <h1 class="product-message">Real Apple Pro</h1>

        <div class="sticky-element section0-canvas">

            <canvas id="main-canvas" width="1920" height="1080"></canvas>

        </div>

        <div class="sticky-element section0-message a">

            <p>온전히 빠져들게 하는 빛깔</p>

        </div>

        <div class="sticky-element section0-message b">

            <p>나의 건강을 채워주는 향기.</p>

        </div>

        <div class="sticky-element section0-message c">

            <p>깨끗한 자연의 신선함.</p>

        </div>

        <div class="sticky-element section0-message d">

            <p>새롭게 입가를 찾아온 매혹</p>

        </div>

    </section>

    <section class="scroll-section" id="section-1">

        <p class="description">

            <strong>진짜 사과 이야기</strong>

            사과를 매일 하나씩 먹으면 의사 볼 일이 없다

            (an apple a day keeps the doctor away)란 속담이 있을 정도로

            유명한 가을 과일 중 하나다. 비슷한 속담이 토마토에도 있다.

            그만큼 비타민C와 무기질이 많아 간식 대용으로 사과를 한 알 먹는다면

            비타민C 부족에 시달릴 일은 없다. 다만 과당이 많아서 통상적인 이미지와는

            달리 다이어트 식품으로서는 그다지 효능이 좋지 않은데, 단위 중량 당 열량은

            아보카도나 바나나 등과 함께 제법 높은 편에 속한다.

            당연히 사과의 크기와 영양상태에 따라 다르지만 300g 정도 나가는 한 알이

            200kcal는 넘을 수 있다고 생각하면 된다.

            비타민 C가 풍부한 만큼 감기 예방에 효과적이나, 한의학적으로 사과는

            차가운 성질을 가지고 있기 때문에 목감기나 기침에 좋지 않다고 한다.

            감기에 좋다고 하는 과일은 배, 그리고 귤도 좋다고 한다.

            배는 과일 중에서도 특히 수분, 무기질이 많은 과일 중 하나이기 때문에

            감기에 걸렸을 때 수분 보충에 아주 적절한 식품이다.

        </p>

    </section>

    <footer>

        Copyleft 2022 EZEN Sangbong Allright reserved.

    </footer>

    <script src="./main.js"></script>

</body>

</html>

@charset 'utf-8';

html {

    font-size: 14px;

    font-family: 'Noto Sans KR', sans-serif;

    color: white;

}

body {

    overflow-x: hidden;

    letter-spacing: -0.05em;

}

a {

    color: white;

    text-decoration: none;

}

footer {

    display: flex;

    height: 5em;

    justify-content: center;

    align-items: center;

    font-weight: bold;

    font-size: 2rem;

    background-color:gray;

    color: black;

}

.global-nav {

    position: absolute;

    z-index: 10;

    width: 100%;

    height: 44px;

    left: 0;

    top: 0;

}

.local-nav {

    position: absolute;

    z-index: 10;

    width: 100%;

    height: 52px;

    left: 0;

    top : 45px;

}

.global-nav-links {

    display: flex;

    align-items: center;

    justify-content: space-between;

    height: 100%;

    max-width: 1000px;

    margin: 0 auto;

}

.local-nav-links {

    display: flex;

    align-items: center;

    justify-content: right;

    height: 100%;

    max-width: 1000px;

    margin: 0 auto;

    border-bottom: 1px solid gray;

}

.local-nav-product-link {

    font-size: 1.5rem;

    font-weight: bold;

    margin-right: auto;

}

.local-nav-link {

    margin-left: 2em;

}

.product-message {

    position: relative;

    z-index: 9;

    font-size: 7rem;

    text-align: center;

    margin-top: 1.5em;

}

.description {

    position: relative;

    font-size: 2rem;

    z-index: 9;

    line-height: 3em;

    color:rgb(175, 175, 175);

    max-width: 1000px;

    margin: 0 auto;

}

.description > strong {

    float: left;

    font-size: 4rem;

    color: white;

    margin-right: 20px;

}

.section0-message

{

    opacity: 0;

    font-size: 4rem;

    font-weight: bold;

    width: 100%;

    height: 6rem;

    top: 6em;

    text-align: center;

}

.section0-canvas

{

    left: 0;

    top: 0;

    width: 100%;

    background-color: black;

}

.sticky-element

{

    display: none;

    position: fixed;

}

#show-section0 .sticky-element

{

    display: block;

}

.local-nav-sticky .local-nav{

    position: fixed;

    left: 0;

    top: 0;

    border-bottom: 1px solid gray;

    -webkit-backdrop-filter: saturate(180%) blur(15px);

    -moz-backdrop-filter: saturate(180%) blur(15px);

    -o-backdrop-filter: saturate(180%) blur(15px);

    backdrop-filter: saturate(180%) blur(15px);

}

.local-nav-sticky .local-nav-links {

    border: none;

}

.scroll-section {

    background-color: black;

}

(()=>{

    let yOffset = 0;            // 전체 문서에서의 yOffset

    let sectionYOffset = 0;     // 섹션내에서의 yOffset

    let currentSection = 0;     // 현재 섹션 번호

    const sectionSet = [

        {

            height: 0,

            hMultiple: 5,

            objs : {

                container : document.querySelector('#section-0'),

                messageA : document.querySelector('.section0-message.a'),

                messageB : document.querySelector('.section0-message.b'),

                messageC : document.querySelector('.section0-message.c'),

                messageD : document.querySelector('.section0-message.d'),

                canvas : document.querySelector('#main-canvas'),

                context : document.querySelector('#main-canvas').getContext('2d'),

                canvasImages : [],

            },

            vals : {

                imageCount              : 600,

                imageSequence           : [0, 599],

                canvas\_opacity\_out      : [1, 0, {start: 0.71, end:0.99}],

                messageA\_opacity\_in     : [0, 1, {start: 0.09, end: 0.19}],

                messageA\_opacity\_out    : [1, 0, {start: 0.21, end: 0.29}],

                messageA\_translateY\_in  : [0, -15, {start: 0.09, end: 0.19}],

                messageA\_translateY\_out : [-15, -30, {start: 0.21, end: 0.29}],

                messageB\_opacity\_in     : [0, 1, {start: 0.31, end: 0.39}],

                messageB\_opacity\_out    : [1, 0, {start: 0.41, end: 0.49}],

                messageB\_translateY\_in  : [0, -15, {start: 0.31, end: 0.39}],

                messageB\_translateY\_out : [-15, -30, {start: 0.41, end: 0.49}],

                messageC\_opacity\_in     : [0, 1, {start: 0.51, end: 0.59}],

                messageC\_opacity\_out    : [1, 0, {start: 0.61, end: 0.69}],

                messageC\_translateY\_in  : [0, -15, {start: 0.51, end: 0.59}],

                messageC\_translateY\_out : [-15, -30, {start: 0.61, end: 0.69}],

                messageD\_opacity\_in     : [0, 1, {start: 0.71, end: 0.79}],

                messageD\_opacity\_out    : [1, 0, {start: 0.81, end: 0.85}],

                messageD\_translateY\_in  : [0, -15, {start: 0.71, end: 0.79}],

                messageD\_translateY\_out : [-15, -30, {start: 0.81, end: 0.85}],

            }

        },

        {

            height: 0,

            hMultiple: 2,

            objs : {

                container : document.querySelector('#section-1'),

            },

            vals : {

            }

        }

    ];

    // section의 크기, 위치등의 레이아웃을 설정한다.

    const setLayout = function()

    {

        // section의 높이를 설정한다.

        for (let i = 0; i < sectionSet.length; i++)

        {

            sectionSet[i].height = window.innerHeight \* sectionSet[i].hMultiple;

            sectionSet[i].objs.container.style.height = `${sectionSet[i].height}px`;

        }

    }

    // 현재 섹션을 알아내는 함수.

    const getCurrentSection = function()

    {

        let section = 0;

        if (yOffset <= sectionSet[0].height)

        {

            section = 0;

        }

        else if ((yOffset > sectionSet[0].height) &&

                 (yOffset <= sectionSet[0].height + sectionSet[1].height))

        {

            section = 1;

        }

        else

        {

        }

        return section;

    }

    // body태그에 아이디를 넣어주는 함수

    const setBodyID = function(section)

    {

        document.body.setAttribute('id', `show-section${section}`);

    }

    const setLocalnavMenu = function(offset)

    {

        if (offset > 44)

        {

            document.body.classList.add('local-nav-sticky');

        }

        else

        {

            document.body.classList.remove('local-nav-sticky');

        }

    }

    const getPrevSectionHeight = function()

    {

        let prevHeight = 0;

        for (let i = 0; i < currentSection; i++)

        {

            prevHeight = prevHeight + sectionSet[i].height;

        }

        return prevHeight;

    }

    //현재 섹션에서 적용해야할 CSS값을 계산하는 함수.

    const calcValue = function(values)

    {

        let result;

        const height = sectionSet[currentSection].height;

        let rate;

        let partStart;      // 실제 start값

        let partEnd;        // 실제 end값

        let partHeight;     // 높이.

        if (values.length == 2)

        {

            rate = sectionYOffset / height;

            result = (rate \* (values[1] - values[0])) + values[0];

        }

        // start, end값이 들어온 경우. [0, 1, {start: 0.1, end: 0.2}],

        else if (values.length === 3)

        {

            partStart = values[2].start \* height;

            partEnd   = values[2].end \* height;

            partHeight = partEnd - partStart;

            if (sectionYOffset < partStart)

            {

                result = values[0];

            }

            else if (sectionYOffset > partEnd)

            {

                result = values[1];

            }

            else

            {

                rate = (sectionYOffset - partStart) / partHeight;

                result = (rate \* (values[1] - values[0])) + values[0]

            }

        }

        return result;

    }

    const loadCanvasImages = function()

    {

        let imageElement;

        let imageCount = sectionSet[0].vals.imageCount;

        for (let i = 0; i < imageCount; i++)

        {

            imageElement = new Image();

            imageElement.src = `../image/apple\_${i}.png`;

            sectionSet[0].objs.canvasImages.push(imageElement);

        }

        imageElement.addEventListener('load', ()=>{

            sectionSet[0].objs.context.drawImage(sectionSet[0].objs.canvasImages[0], 0, 0);

        });

    }

    const playAnimation = function()

    {

        const values  = sectionSet[currentSection].vals;

        const objects = sectionSet[currentSection].objs;

        const scrollRate = sectionYOffset / sectionSet[currentSection].height;

        let transY = 0;

        let imageIndex = 0;

        switch(currentSection)

        {

            case 0 :

                imageIndex = Math.round(calcValue(values.imageSequence));

                objects.context.drawImage(objects.canvasImages[imageIndex], 0, 0);

                if ((scrollRate > 0.7) && (scrollRate < 1))

                {

                    objects.canvas.style.opacity = calcValue(values.canvas\_opacity\_out);

                }

                objects.messageA.style.opacity = 0;

                objects.messageB.style.opacity = 0;

                objects.messageC.style.opacity = 0;

                objects.messageD.style.opacity = 0;

                if (scrollRate <= 0.2)

                {

                    // messageA

                    // [0, 1, {start: 0.09, end: 0.19}],

                    objects.messageA.style.opacity = calcValue(values.messageA\_opacity\_in);

                    transY = calcValue(values.messageA\_translateY\_in);

                    objects.messageA.style.transform = `translateY(${transY}%)`;

                }

                else if ((scrollRate > 0.2) && (scrollRate <= 0.3))

                {

                    //[1, 0, {start: 0.21, end: 0.29}],

                    objects.messageA.style.opacity = calcValue(values.messageA\_opacity\_out);

                    transY = calcValue(values.messageA\_translateY\_out);

                    objects.messageA.style.transform = `translateY(${transY}%)`;

                }

                else if ((scrollRate > 0.3) && (scrollRate <= 0.4))

                {

                    // [0, 1, {start: 0.31, end: 0.39}]

                    objects.messageB.style.opacity = calcValue(values.messageB\_opacity\_in);

                    transY = calcValue(values.messageB\_translateY\_in);

                    objects.messageB.style.transform = `translateY(${transY}%)`;

                }

                else if ((scrollRate > 0.4) && (scrollRate <= 0.5))

                {

                    // [1, 0, {start: 0.41, end: 0.49}],

                    objects.messageB.style.opacity = calcValue(values.messageB\_opacity\_out);

                    transY = calcValue(values.messageB\_translateY\_out);

                    objects.messageB.style.transform = `translateY(${transY}%)`;

                }

                else if ((scrollRate > 0.5) && (scrollRate <= 0.6))

                {

                    // [0, 1, {start: 0.51, end: 0.59}]

                    objects.messageC.style.opacity = calcValue(values.messageC\_opacity\_in);

                    transY = calcValue(values.messageC\_translateY\_in);

                    objects.messageC.style.transform = `translateY(${transY}%)`;

                }

                else if ((scrollRate > 0.6) && (scrollRate <= 0.7))

                {

                    // [1, 0, {start: 0.61, end: 0.69}]

                    objects.messageC.style.opacity = calcValue(values.messageC\_opacity\_out);

                    transY = calcValue(values.messageC\_translateY\_out);

                    objects.messageC.style.transform = `translateY(${transY}%)`;

                }

                else if ((scrollRate > 0.7) && (scrollRate <= 0.8))

                {

                    // [0, 1, {start: 0.71, end: 0.79}]

                    objects.messageD.style.opacity = calcValue(values.messageD\_opacity\_in);

                    transY = calcValue(values.messageD\_translateY\_in);

                    objects.messageD.style.transform = `translateY(${transY}%)`;

                }

                else if ((scrollRate > 0.8) && (scrollRate <= 0.9))

                {

                    // [1, 0, {start: 0.81, end: 0.85}

                    objects.messageD.style.opacity = calcValue(values.messageD\_opacity\_out);

                    transY = calcValue(values.messageD\_translateY\_out);

                    objects.messageD.style.transform = `translateY(${transY}%)`;

                }

                break;

            case 1 :

                // 1번 섹션의 애니메이션 처리

                break;

        }

    }

    // 스크롤시 제일 마지막에 처리할 기능들...

    const scrollProc = function()

    {

        playAnimation();

    }

    ///////////////////////////////////////////////////////////////////////////////

    // Event Handler

    // 스크롤 이벤트

    window.addEventListener('scroll', ()=>{

        yOffset = window.scrollY;

        currentSection = getCurrentSection();

        sectionYOffset = yOffset - getPrevSectionHeight();

        setBodyID(currentSection);

        setLocalnavMenu(yOffset);

        scrollProc();

    });

    // 처음 로딩시 (리소스가 모두 로딩될 때)

    window.addEventListener('load', ()=>{

        setLayout();

        loadCanvasImages();

        currentSection = getCurrentSection();

        sectionYOffset = yOffset - getPrevSectionHeight();

        setBodyID(currentSection);

        setLocalnavMenu(yOffset);

    });

    // 윈도우의 크기가 바뀔 때

    window.addEventListener('resize', ()=>{

        setLayout();

        currentSection = getCurrentSection();

        sectionYOffset = yOffset - getPrevSectionHeight();

        setBodyID(currentSection);

        setLocalnavMenu(yOffset);

    });

})();