Senkron ve Asenkron nedir?

Senkron ve asenkron JavaScript (JS), JavaScript programlarının çalışma şekillerini ifade eder.

Senkron JavaScript, işlemlerin sırayla ve adım adım gerçekleştirildiği bir yaklaşımdır. Bir işlem tamamlanmadan bir sonraki işleme geçilmez. Örneğin, bir fonksiyon çağrısı yapılırsa, bu fonksiyon tamamlanana kadar diğer işlemler beklemek zorundadır. Senkron JavaScript, basit programlarda kullanılabilecek doğrudan bir yaklaşımdır.

Asenkron JavaScript ise işlemleri eşzamansız olarak yürütür. Bu, bir işlem başlatıldıktan sonra, sonucun tamamlanmasını beklemek yerine diğer işlemlere devam edebileceği anlamına gelir. Asenkron işlemler genellikle zaman alan veya harici kaynaklarla etkileşime geçen işlemler için kullanılır. Örneğin, bir dosyanın okunması veya bir ağ isteği gibi işlemler asenkron olarak gerçekleştirilebilir. Bu tür işlemler, tamamlanması zaman alan süreçler olduğunda programın beklemesini engeller ve diğer işlemlere geçmesine izin verir. Asenkron JavaScript, JavaScript'in olay tabanlı modeline dayanan bir yapı kullanır ve genellikle geri çağırma fonksiyonları, Promise'ler veya async/await yapıları kullanılarak ifade edilir.

Asenkron JavaScript, daha karmaşık ve verimli programlar geliştirmek için yaygın olarak kullanılır. Özellikle web tarayıcılarında, ağ istekleri ve kullanıcı etkileşimleri gibi olaylar asenkron olarak ele alınır. Bu sayede kullanıcı arayüzü etkileşimli kalırken, arka planda uzun süren işlemler gerçekleştirilebilir. Asenkron yapılar, beklemeleri en aza indirerek programların daha hızlı ve verimli çalışmasına yardımcı olur.

Binary Octal Decimal Hexadecimal nedir?

Binary (İkili): İkili sistem, yalnızca "0" ve "1" rakamlarını kullanarak sayıları temsil etmek için kullanılan bir sayı sistemidir. Bilgisayarların temel işlem mantığı ve elektronik devrelerde sıkça kullanılır. İkili sistemdeki her basamak, bir gücünün (2 üzeri) ifadesini temsil eder. Örneğin, 1010 ikili sayısı, 1x2^3 + 0x2^2 + 1x2^1 + 0x2^0 olarak hesaplanır ve 10 onluk sayısına denk gelir.

Octal (Sekizli): Sekizli sistem, "0" ile "7" arasındaki rakamları kullanarak sayıları temsil etmek için kullanılan bir sayı sistemidir. Her bir sekizlik basamak, bir gücünün (8 üzeri) ifadesini temsil eder. Örneğin, 25 sekizli sayısı, 2x8^1 + 5x8^0 olarak hesaplanır ve 21 onluk sayısına denk gelir.

Decimal (Onlu): Onlu sistem, günlük hayatta en yaygın olarak kullandığımız sayı sistemidir. Rakamlar 0 ile 9 arasındadır ve her bir onluk basamak, bir gücünün (10 üzeri) ifadesini temsil eder. Örneğin, 348 onluk sayısı, 3x10^2 + 4x10^1 + 8x10^0 olarak hesaplanır.

Hexadecimal (Onaltılı): Onaltılı sistem, "0" ile "9" arasındaki rakamlara ek olarak "A" ile "F" arasındaki harfleri (A, B, C, D, E, F) kullanarak sayıları temsil etmek için kullanılan bir sayı sistemidir. Her bir onaltılı basamak, bir gücünün (16 üzeri) ifadesini temsil eder. Örneğin, 1A hexadecimal sayısı, 1x16^1 + 10x16^0 olarak hesaplanır ve 26 onluk sayısına denk gelir.

Bu dört sayı sistemi, bilgisayar bilimi, programlama, dijital elektronik ve veri temsili gibi alanlarda önemli bir rol oynar. Bilgisayarlar ve diğer dijital cihazlar, verileri ikili (binary) şekilde temsil ederken, programlama dilleri genellikle onlu (decimal) sistemle çalışırken, hex (hexadecimal) sistem genellikle bellek adresleri, renk kodları ve diğer bazı alanlarda kullanılır. Dönüşüm yöntemleri ve matematiksel ilişkiler aracılığıyla bu farklı sayı sistemleri arasında dönüşüm yapmak mümkündür.

Number() ve ParseInt Arasındaki Fark

Number() ve parseInt() fonksiyonları, bir metin veya değeri sayıya dönüştürmek için kullanılan iki farklı JavaScript yöntemidir. İşlevleri ve kullanımları arasında bazı farklılıklar vardır:

Number(): Bu JavaScript yöntemi, verilen değeri sayıya dönüştürmek için kullanılır. İlgili parametreyi alır ve dönüştürülen sayıyı döndürür. İşte Number() fonksiyonunun bazı özellikleri:

- Boşluk karakterleri otomatik olarak kaldırılır.
- Eğer verilen parametre tam sayı veya ondalık sayı ise, bunu doğrudan dönüştürür.
- Eğer verilen parametre bir sayısal ifade içeriyorsa, o sayısal ifadeyi dönüştürür. Örneğin, "123" ifadesini sayıya dönüştürür.
- Eğer verilen parametre sayısal bir ifade içermiyorsa, NaN (Not-a-Number) değerini döndürür.

ParseInt(): Bu JavaScript yöntemi, bir metni belirli bir sayı tabanına göre sayıya dönüştürmek için kullanılır. İlgili parametreyi ve tabanını alır ve dönüştürülen sayıyı döndürür. İşte parseInt() fonksiyonunun bazı özellikleri:

- Boşluk karakterleri otomatik olarak kaldırılır.
- Eğer verilen parametre tam sayı veya ondalık sayı ise, tam sayı kısmını döndürür.
- Eğer verilen parametre bir sayısal ifade içermiyorsa, NaN (Not-a-Number) değerini döndürür.
- İkinci bir parametre olarak taban belirtilmezse, varsayılan olarak 10 tabanını kullanır. Ancak ikinci bir parametre olarak taban belirtilirse, metini belirtilen tabanda sayıya dönüştürür.

Özetlemek gerekirse, Number() fonksiyonu genel olarak bir değeri sayıya dönüştürmek için kullanılırken, parseInt() fonksiyonu metin tabanlı dönüşümler ve belirli bir taban kullanma özelliği ile daha spesifik bir amaç için kullanılır.

String() ve .toString() Arasındaki Fark

String() ve .toString() JavaScript yöntemleri, bir değeri metin (string) türüne dönüştürmek için kullanılan iki farklı yöntemdir. İşlevleri ve kullanımları arasında bazı farklılıklar vardır:

String(): Bu JavaScript yöntemi, verilen bir değeri metin (string) türüne dönüştürmek için kullanılır. İlgili parametreyi alır ve dönüştürülen metin değerini döndürür. İşte String() yönteminin bazı özellikleri:

- Eğer verilen değer bir nesne ise, .toString() yöntemini çağırarak nesneyi metin türüne dönüştürür.
- Eğer verilen değer null veya undefined ise, "null" veya "undefined" metinini döndürür.
- Eğer verilen değer boolean bir değer ise, "true" veya "false" metinini döndürür.
- Eğer verilen değer sayısal bir değer ise, sayıyı metin olarak döndürür.
-

.toString(): Bu JavaScript yöntemi, bir değeri metin (string) türüne dönüştürmek için kullanılan bir yöntemdir. İlgili değeri alır ve dönüştürülen metin değerini döndürür. İşte .toString() yönteminin bazı özellikleri:

- .toString() yöntemi, nesnelerin prototipinde tanımlanan bir yöntemdir. Bu nedenle, bir nesne üzerinde doğrudan çağrılabilir.
- Eğer verilen değer bir nesne ise, nesnenin .toString() yöntemini çağırarak nesneyi metin türüne dönüştürür.
- Eğer verilen değer null veya undefined ise, TypeError hatası alırsınız.
- Eğer verilen değer sayısal bir değer ise, sayıyı metin olarak döndürür.

Özetlemek gerekirse, String() yöntemi bir değeri metin türüne dönüştürmek için kullanılırken, .toString() yöntemi nesnelerin prototipinde tanımlanan bir yöntem olup, bir nesneyi metin türüne dönüştürmek için kullanılır.